Navigational Strategies for Control of Underwater Robot using AI based Algorithms

Shubhasri Kundu

Navigational Strategies for Control of Underwater Robot using AI based Algorithms

Thesis Submitted to the
Department of Mechanical Engineering
National Institute of Technology, Rourkela
For award of the degree

of

Doctor of Philosophy

by

Shubhasri Kundu

under the guidance of

Prof. Dayal R. Parhi

Department of Mechanical Engineering
National Institute of Technology Rourkela
Orissa (India)-769008

November 2015

Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgement has been made in the text.

(*Shubhasri Kundu*)

Date:

Certificate

This is to certify that the thesis entitled, ***"Navigational Strategies for Control of Underwater Robot using AI based Algorithms",*** *being submitted by Mrs. Shubhasri Kundu to the Department of Mechanical Engineering, National Institute of Technology, Rourkela, for the partial fulfilment of award of the degree Doctor of Philosophy, is a record of bona fide research work carried out by him under my supervision and guidance.*

This thesis in my opinion, is worthy of consideration for award of the degree of Doctor of Philosophy in accordance with the regulation of the institute. To the best of my knowledge, the results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Supervisor

Date:

(Dr.*Dayal R. Parhi*)
Professor, Mechanical Engineering
Department of Mechanical Engineering
National Institute of Technology, Rourkela,
Orissa, India- 769008.

Acknowledgements

My first thanks are to the Almighty God, without whose blessings, I wouldn't have been writing this “acknowledgments".

I would like to extend my heartfelt indebtedness and gratitude to Prof. Dayal R. Parhi for his kindness in providing me an opportunity to work under his supervision and guidance. During this period, without his endless efforts, immense knowledge, deep patience, invaluable guidance and answers to my numerous questions, this research would have never been possible. I am especially obliged to him for teaching me both research and writing skills, which have been proven beneficial for my current research and future career. He showed me different ways to approach a research problem and the need to be persistent to accomplish any goal. It has been a great honour and pleasure for me to do research under the supervision of Dr. Dayal R. Parhi.

I am thankful to Prof. Sunil Kumar Sarangi, Director of National Institute of Technology, for giving me an opportunity to be a part of this institute of national importance and to work under the supervision of Prof. Dayal R. Parhi. I am sincerely obliged to Prof. S. S. Mahapatra, Head of the Department, Department of Mechanical Engineering, for providing me all official and laboratory facilities during research period. His incessant encouragement towards research work has inspired me a lot.

I express my gratitude to Prof. P.K Roy, Chairman DSC and DSC members for their indebted help and valuable suggestions for accomplishment of dissertation.

I thank all the members of the Department of Mechanical Engineering, and the Institute, who helped me in various ways towards the completion of my work.

I would like to thank all my friends and lab-mates for their encouragement and understanding. Their support and lots of lovely memory with them can never be captured in words. I want to specially thank Mr. Bijan Sethi and Mr. Maheshwar Das, for helping me in various ways throughout my Ph.D work.

Finally, I thank my parents, husband and entire family members for their unlimited support and strength. Without their dedication and dependability, I could not have pursued my Ph.D degree at the National Institute of Technology Rourkela.

Shubhasri Kundu

Synopsis

Autonomous underwater robots have become indispensable marine tools to perform various tedious and risky oceanic tasks of military, scientific, civil as well as commercial purposes. To execute hazardous naval tasks successfully, underwater robot needs an intelligent controller to manoeuver from one point to another within unknown or partially known three-dimensional environment.

This dissertation has proposed and implemented various AI based control strategies for underwater robot navigation. Adaptive versions of neuro-fuzzy network and several stochastic evolutionary algorithms have been employed here to avoid obstacles or to escape from dead end situations while tracing near optimal path from initial point to destination of an impulsive underwater scenario. A proper balance between path optimization and collision avoidance has been considered as major aspects for evaluating performances of proposed navigational strategies of underwater robot. Online sensory information about position and orientation of both target and nearest obstacles with respect to the robot's current position have been considered as inputs for path planners. To validate the feasibility of proposed control algorithms, numerous simulations have been executed within MATLAB based simulation environment where obstacles of different shapes and sizes are distributed in a chaotic manner. Simulation results have been verified by performing real time experiments of robot in underwater environment. Comparisons with other available underwater navigation approaches have also been accomplished for authentication purpose. Extensive simulation and experimental studies have ensured the obstacle avoidance and path optimization abilities of proposed AI based navigational strategies during motion of underwater robot. Moreover, a comparative study has been performed on navigational performances of proposed path planning approaches regarding path length and travel time to find out most efficient technique for navigation within an impulsive underwater environment.

Keywords: Underwater robot, Three-dimensional Navigation, ANFIS, Shuffled Frog Leaping Algorithm, Differential Evolution, Harmony Search

Contents

List of Figures

List of Tables

Abbreviations

AUV : Autonomous Underwater Vehicle

ROV : Remotely Operated Vehicle

GNC : Guidance, Navigation and Control

SNAME : Society of Naval Architects and Marine Engineers

UUV : Unmanned Underwater Vehicle

DOF : Degree of Freedom

3D : Three-dimension

ANFIS : Adaptive Neuro-Fuzzy Inference System

NOD : Nearest Obstacle Distance

TD : Target Distance

angobs(h) : Heading angles between robot and its nearest obstacles in the horizontal plane

angobs (v) : Heading angles between robot and its nearest obstacles in the vertical plane

angtar(h) : Angle between robot and target in the horizontal plane

angtar(v) : Angle between robot and target in the vertical plane

OA : Obstacle Avoidance

TS : Target seeking

BP : Back Propagation

GD : Gradient Descent

LSE : Least Square Estimation

RLS : Recursive Least Square Estimation

RMSE : Root Mean Square Error

EA : Evolutionary Algorithm

MA : Memetic Algorithms

GA : Genetic Algorithm

SFLA : Shuffled Frog Leaping Algorithm

DE : Differential Evolution

HS : Harmony search

List of Symbols

{E}	Earth Fixed Frame or Inertial Frame
{B}	Body Fixed Frame
η	Vector for representing Underwater Robot's Position and Orientation in three-dimensional workspace
v_c	Vector for denoting Linear Velocity
ω	Vector for denoting Angular Velocity
${}^{E}R_{B}(\psi,\theta,\phi)$	Rotation matrix for Linear Velocity Transformation
${}^{E}W_{B}(\phi,\theta)$	Rotation matrix for Angular Velocity Transformation
$V(\eta,\dot{\eta})$	Lypunov Candidate Function
$\{robx_i, roby_i, robz_i\}$	Underwater robot's position in i^{th} iteration
K_{oa}	Weight factor for Obstacle Avoidance behavioral module
K_{ts}	Weight factor for Target Seeking behavioral module
E_p	Error measure for the p^{th} training pattern
$\Delta\psi$	Deviation in heading angle for horizontal plane
$\Delta\theta$	Deviation in heading angle for vertical plane
$Fitness_{p,i}$	Fitness of p^{th} population member in i^{th} iteration
λ_{ls}	Scaling constant associated with the distance between robot's current pose and its next probable position in the definition of Fitness Function
λ_{gd}	Scaling constant associated with the distance between robot and goal position in the definition of Fitness Function
λ_{nod}	Scaling constant associated with the distance between robot and nearest obstacle in the definition of Fitness Function
rand (0,1)	Random value between 0 and 1
$U(0,1)$	Uniform Distribution between 0 and 1
F_i	Scaling factor for Difference Vector in i^{th} iteration of Differential Evolution Algorithm
CR_i	Crossover constant in i^{th} iteration of Differential Evolution Algorithm

1. Introduction

Autonomous underwater vehicles (AUVs) have become indispensable naval devices for saving human resources from being exploited in tedious and risky oceanic tasks of various military, scientific, civil as well as commercial applications [1-3]. The autonomous control of underwater vehicles poses serious challenges due to the AUVs' dynamics. AUVs dynamics are highly nonlinear with respect to time and the hydrodynamic coefficients of vehicles are difficult to estimate accurately because of the variations of these coefficients with different navigation conditions and external disturbances. Therefore, research on navigational strategies of underwater robot without any modelling of robot or mapping of environment has received huge interests.

1.1 Background & Motivation

Massive advances in technology as well as human attraction towards autonomous vehicle results in easy access to risk prone areas of world (air, surface or underwater). Such vehicles must be responsive, adaptable, robust and also multitasking by nature. Now-a-days, growth in researches on unmanned autonomous underwater robots leads towards imminent deployment of undersea circumstances for various commercial, scientific, defense and academic applications like long term observations, exploration of mines from seabed, surveillance of ice-covered areas of ocean, taking videos of rare views of sea floor, retrieval of ship wreckages in deep ocean and security of aqua lives etc. While performing such precarious marine tasks, underwater robot needs autonomous control system to manoeuvre from one point to another within workspace without any human intervention [4]. The locations of these waypoints must be optimized to get a path of minimum length by connecting them [5]. One major constraint on path planning of underwater robot is to search a safe path while moving from a start point to a destination point without collisions with obstacles. Enormous research concerns about underwater path planning strategies have come forward during past several decades.

Generally, AUV has several available paths to complete a given task, an optimal (or near optimal) path has to be chosen under a certain criterion in practical applications. Underwater robot faces much more intricacies than ground robots due to high water resistance and low bandwidth communication of marine environment. 3D path planning is a classical optimization problem. Most classical navigational algorithm focuses on

detailed map of environment using expensive hardware such as laser scanner, telescopes, camera etc. These methods may create computational burden to the processing units and hazardous for real life applications. Many researchers have applied new optimization algorithms to solve the path planning problem in recent years. Present research work has been motivated by the success of evolutionary optimization in navigation of underwater robot. Several intelligent methods have been approached here to achieve high degree of autonomy during underwater navigation.

1.2 Aims and Objectives

The concern of present research work is to find out the robust and accurate navigational methodology for underwater robot which can trace near optimal collision free path in an autonomous way while moving from start to goal point within partially known or completely unknown marine environment.

The major objectives of present research work can be briefly outlined as follows:

(a) Integration of ultrasonic sensors and their controller within underwater robot for perceiving the environment.

(b) Incorporation of image sensor for mapping of underwater surrounding and detection of submerged target.

(c) Measurement of the depth of underwater robot for its current location by integrating of on-board sonar depth sensor.

(d) Development of ANFIS, SFLA, DE, HS and hybrid algorithms for navigation of underwater robot from source to goal avoiding obstacles.

1.3 Methodologies

To carry out underwater robot from start to goal point within marine environment, the following methodologies have been adapted:

- Kinematic and dynamic modelling of small size underwater robot in a simplified manner along with the stability analysis based on certain assumptions.
- To perform underwater tasks in an autonomous way ultrasonic, image and depth sensors are externally mounted on underwater robot. Control algorithms are embedded within inbuilt microcontroller of underwater robot. Specifications for sensors and controller have been illustrated in Appendix-A.

- Design and development of navigational controller for underwater robot using Manifold adaptive neuro-fuzzy inference system (ANFIS) to achieve preliminary reactive robotic behaviors (Obstacle avoidance and Target seeking).
- Implementation of Shuffled frog leaping algorithm (SFLA) based global and local search approach for three-dimensional path optimization.
- Autonomous navigation of underwater robot by employing deterministic behavior of Differential Evolution (DE) approach.
- Development of navigational controller for underwater robot based on global optimization ability of Harmony Search (HS) algorithm.
- Hybridization of DE and HS approaches to balance intensification and diversification of three-dimensional path optimization process.
- Training of ANFIS models with SFLA-RLS based hybrid learning approach to control the motion direction of underwater robot with high degree of accuracy during autonomous navigation.

1.4 Novelty of Thesis

Present research work has introduced novel adaptation mechanism for evolutionary algorithms like Shuffled frog leaping algorithm (SFLA), Differential Evolution (DE) and Harmony Search (HS) to achieve proper balance between exploration and exploitation abilities of search process. Proposed new versions of metaheuristic approaches have been successfully implemented as obstacle avoidance and path optimization strategies of underwater robot. A new hybrid learning approach has been proposed for training of ANFIS models. Hybridization of evolutionary algorithms like DE and HS has also been employed as navigational strategy. Navigational performances of all proposed approaches have been verified in numerous simulation and experimental scenarios for underwater navigation.

1.5 Framework of Dissertation

The investigations carried out in present dissertation can be approximately divided into ten chapters which are outlined below:

Chapter 1 has provided an introduction to recent applications of underwater robots and discusses about the motivation behind current research work. An overview on major goals of the present work has also been narrated within this chapter. Novelty of recent research work has also been mentioned.

Chapter 2 has presented the literature review on modeling of underwater robot and its various navigational approaches.

Chapter 3 has demonstrated kinematic and dynamic modelling of underwater robot and its stability issues.

Chapter 4 has employed manifold ANFIS approach to estimate the change in heading angle of underwater robot for avoiding obstacles during navigation.

Chapter 5 has proposed a new adaptive version of memetic evolution based optimization method (Shuffled Frog Leaping Algorithm) to achieve the near optimal safe path during underwater navigation.

Chapter 6 has focused on effective local search behavior of Differential Evolution to enhance convergence speed of three-dimensional path optimization method.

Chapter 7 has addressed Harmony Search based global optimization method for tracing near optimal three-dimensional path.

Chapter 8 has hybridized two metaheuristics techniques (i.e. DE and HS) of different benefits to enhance accuracy in navigational performance of underwater robot.

Chapter 9 has proposed a hybrid learning approach for training of ANFIS parameters to gain more accuracy while regulating heading angle of underwater robot.

Chapter 10 has compared the performances of all proposed navigational strategies for different simulation and experimental scenarios.

Chapter 11 has drawn the conclusion of present dissertation along with some future aspects of present research work.

2. Literature Review

Current investigation has applied enormous effort to analyze the various path planning and control strategies for underwater navigation which have been developed so far. A brief overview of extensive research work of last few decades on modelling of underwater robot and its navigational strategies has been elaborated in this chapter.

2.1 Introduction

The ocean covers about 70% of the earth surface and has great effect on the future existence of all human beings, beside the land and aerospace [6]. Underwater robotics is no doubt an important scientific area due to its great applications that vary from scientific research of ocean, inspection of undersea facilities to military operations. Presently, the seas represent critical sources of food and other resources such as oil and gas. Unmanned underwater vehicles classified into two categories: remotely operated vehicles (ROV) which are tethered one and Autonomous Underwater Vehicles (AUV) which are completely autonomous vehicle [7]. Offshore oil and gas installations are presently serviced almost exclusively by remotely operated vehicles (ROVs) physically connected via a tether to receive power and data, with human divers used only for the shallowest installations. The effectiveness of using ROVs decreases with depth mainly due to the cost increase and the difficulties of handling the long tether.

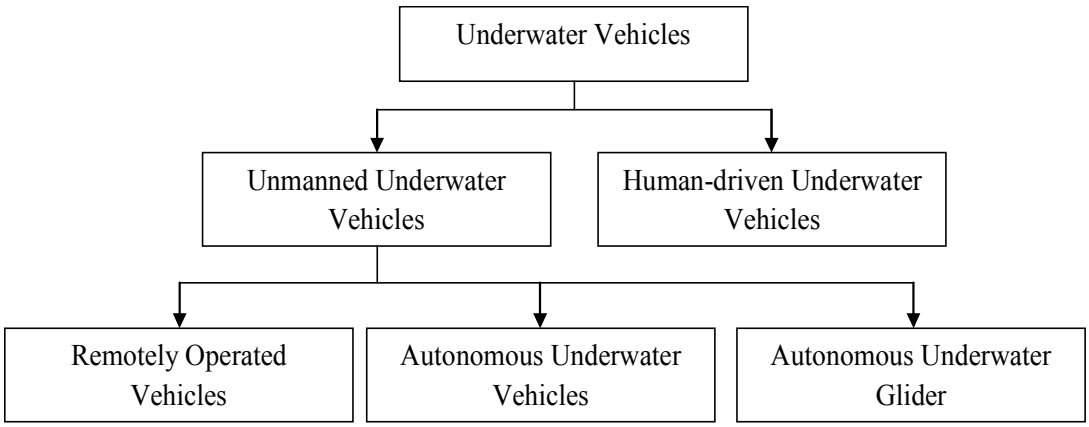


Figure 2.1: Categorization of Underwater Vehicles

Autonomous underwater vehicles (AUVs) are free swimming unoccupied underwater vehicles that can overcome the limitations imposed by ROV tethers for some tasks. Such vehicles carry their own energy supplies (presently batteries, perhaps fuel cells in the

future) and communicate only through acoustics and perhaps optical links in the near future. Limited communications require these vehicles to operate independently of continuous human control, in many cases the vehicles operate completely autonomously. AUVs are currently used for mine detection and landing site survey, oceanographic sampling, underwater archeology and several manipulation works during long-term undersea surveillance. Today, approximately 200 AUVs are operational, many of them experimental. However, they are maturing rapidly. AUVs must accomplish complex tasks and diverse missions while maintaining stable physical control with six spatial degrees of freedom. When compared to indoor, ground, airborne or space environments, the underwater domain typically imposes the most restrictive physical control and sensor limitations upon a robot.

2.2 Autonomous Underwater Robot Model

With respect to vehicles in other environments, underwater vehicles are probably the hardest to accurately model. It is critical to achieve a correct model but it may ensure that that major faults do not occur with the vehicle [3]. Due to configuration issues, most of AUVs cannot be allowed to move in pitch or roll direction where the angle is very steep. The instability in AUV configuration may results in the potential damage to internal components. An incorrect model could allow this to occur as well as allowing a number of other failures to occur in control, navigation or power which can result in damage or even loss of the entire vehicle. AUV design must provide autonomy, stability and reliability with little tolerance for error. Underwater vehicle dynamics is strongly coupled and highly nonlinear due to added hydrodynamic mass, lift and drag forces acting on the vehicle [8]. In a wide spectrum of applications, underwater vehicles are generally described by nonlinear and time-varying dynamics. For instance, dynamics with variable inertia and buoyancy arriving from sampling missions or hydrodynamics related to large changes of operation velocity or current perturbations in which laminar-to/from-turbulent transitions are involved in the hydrodynamics [9]. Engineering problems associated with the high density, non-uniform and unstructured seawater environment, and the nonlinear response of vehicles make a high degree of autonomy difficult to achieve. Hence six degree of freedom vehicle modelling and simulation are quite important and useful in the development of undersea vehicle control systems [10, 11]. Control systems require

particular attention since closed-form solutions as many hydrodynamics control issues remain unknown while modelling.

Due to the inherent nonlinear equations of motions, perturbed environments and complex missions, subaquatic vehicles require the guidance by means of complex controllers that usually involve automatic speed controls, dynamic positioning and tracking, and autopilot systems for automatic steering of depth and altitude. It is experimentally corroborated that adaptive techniques may provide superior trajectory tracking performance compared with the fixed model-based controllers [12].

The design and development of an autonomous undersea vehicle (AUV) is a complex and expensive task. If the designer relies exclusively on prototype testing to develop the vehicle's geometry and controllers, the process can be lengthy and poses the additional risk of prototype loss. Each design iteration involves changes to the prototype vehicle which may take days and followed by further testing. As a result, designers of AUV's rely increasingly on computer modelling as a design tool, particularly for the initial phases of vehicle development [13]. The trajectory tracking control of autonomous underwater vehicles (AUVs) in six-degrees-of-freedom (6-DOFs) has been analysed by Geranmehr and Nekoo [6]. It has been assumed that the system parameters are unknown and vehicle is under actuated. The desired trajectory must be sufficiently smooth bounded curve parameterized by time or consist of straight line. To guarantee robustness against parameter uncertainty, an adaptive controller based on the Lyapunov's direct method and the back-stepping technique has been proposed while control signals are bounded using saturation functions. Because of the highly nonlinear dynamics and the unpredictable operating environments of AUVs, conventional control schemes such as the PID controller may not be able to provide satisfactory outcomes in relation to the control problems experienced by underwater vehicles. Besides this, it is essential to deal with parametric uncertainties acting on the underwater vehicles in order to obtain a robust autopilot.

The development of a nonlinear control design for a fully-actuated autonomous underwater vehicle (AUV) has been focused by Fischer et al. [14] using a continuous robust integral of the sign of the error control structure to compensate for system uncertainties and sufficiently smooth bounded exogenous disturbances. A Lyapunov stability analysis is included to prove semi-global asymptotic tracking. A state-feedback-

based back-stepping control algorithm has been proposed by Dong et al.[15] to address the point stabilization (or set point regulation) control problem for an under actuated autonomous underwater vehicle (AUV) in the presence of constant and irrotational ocean current disturbance. The proposed backstepping control law for point stabilization has further been enriched by incorporating an additional integral action for enhancing the steady state performance of the AUV control system, while practical asymptotic stability analysis of the system is carried out using Lyapunov theory and Barbalet's Lemma.

Many different adaptive and robust adaptive approaches for underwater vehicles such variety of design techniques based on optimal control, Lyapunov stability theory, feedback linearization, adaptive control, and sliding mode control etc. have been discussed in the literature in the past 20 years to handle uncertainties related to the dynamics, hydrodynamics and external disturbances [6,9,12].

2.3 Guidance and Control

Guidance is the action of determining the course, attitude, and speed of the vehicle, with respect to some reference frame (usually the Earth) [7]. Control is the development and application to a vehicle of appropriate forces and moments for operating point control, tracking, and stabilization. This involves designing the feedforward and feedback control laws. Figure 2.2 has shown the corresponding framework of guidance, control and navigation. The GNC mechanism can be described in detail as follows:

Guidance takes user-defined mission inputs such as waypoints and approach vectors, and converts them into state requests for the controllers (autopilot) [16]. The control system converts the state commands into actuator commands. Navigation feeds back information such as attitude, position and velocity to the guidance and control sub-systems.

The navigation module is usually referred to the on board sensory systems. It comprises the data fusion necessary to locate precisely in 3D the AUV rigid body and the target position. The usual components within the navigation system are a global position system (GPS), an inertial navigation system (INS), a compass, a depth sensor, and others. Thus, the navigation system provides the dynamic mission planner system, the guidance system and the control system with accurate data to achieve their objectives.

The guidance module is frequently associated to a low-level trajectory generation. When the waypoints for the robot are defined, a trajectory to reach them is necessary in order to

feed the controllers set points [7]. One of the most common guidance approaches is based on the generations of way-points. Those are usually stored in a database and are used to generate the vehicle path/trajectory; a passing velocity, in fact, may be defined together with the Cartesian coordinates of the points. The simplest way to connect the way-points is to use the segments connecting two successive way-points. Efficient way-point-based guidance approaches need to take into account the presence of the current and the eventual nonholonomicity of the vehicle [1]. A technique for adaptively tracking bathymetric contours by proper generation of way-points is presented in [17]; environment information is acquired by mean of single vertical sonar. An alternative method is based on line-of-sight guidance [18].

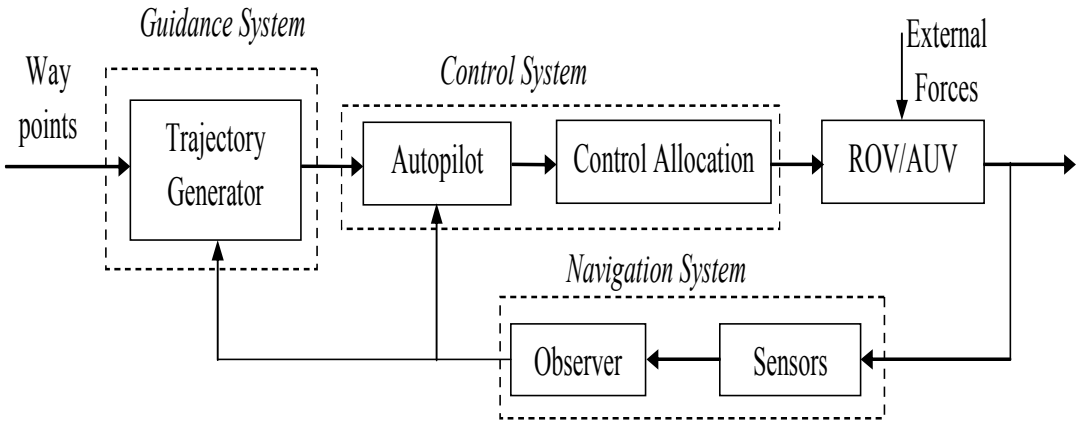


Figure 2.2: Guidance, Navigation and Control for an autonomous marine vehicle [7]

The control module is regarded to the feedback loops allowing the vehicle to describe the trajectory as close as possible to the proposed path given by the guidance module. In effect, assuming that the navigation system yields a clear perception of the AUV's positions, speeds and headings, and the guidance system gives a suitable trajectory to reach a waypoint, there is still remaining a module capable of maintaining the vehicle as close as possible to the prefixed trajectory. Established in this way, the problem to solve at this stage is a control problem to command the vehicle actuators (propellers, rudders and pumps). Control of underwater vehicles needs to consider the different operating conditions and actuating configurations in which a submerged vehicle is required to operate [19]. A mission planner, according to the robot's application, is also necessary to accomplish the task in an autonomous way. A key component of the mission plan is the

path planning. Special sensor acquisitions (snapshots, videos, water samples, and others) or special actions (debris grasping) may also be considered within a mission planner.

Guidance, navigation and control (GNC) are considered to be the three important systems for an autonomous vehicle's utility and mission success. However, considering the autonomous nature of operation, the effectiveness of guidance and navigation of an AUV mainly depends on the real-time path planning [20, 21]. This can be accomplished using any of the available techniques such as road map, cell decomposition, optimal control and potential field methods [22]. Among them, potential field method is a very prominent one for on-line collision avoidance. The potential field method was initially introduced by Khatib [23] in 1986 for developing real-time obstacle avoidance.

A multipoint potential field method (MPPF) for path planning of autonomous underwater vehicles (AUV) in 3D space has been presented by Saravanakumar and Asokan [24]. The algorithm is developed based on potential field method by incorporating a directed search method for sampling the potential field. In this approach, the analytical gradient of the total potential function is not computed, as it is not essentially required for moving the vehicle to the next position. Rather, a hemispherical region in the direction of motion around the AUV's bow is discretized into equiangular points with center as the current position. By determining the point at which the minimum potential exists, the vehicle can be moved towards that point in 3D space. This method is very simple and applicable for real-time implementation. The problem of local minima is also analyzed and found that the local minima in 2D space can be easily overcome with the MPPF. A simple strategy to avoid the local minima in 3D space is also proposed. The proposed method reduces the burden of fine-tuning the positive scaling factors of potential functions to avoid local minimum. Miao and Huang [25] has proposed a novel potential field approach incorporating heuristic obstacle avoidance method for AUV path planning in three dimension environments which have dynamic targets. The approach is able to provide an optimized path solution for the dynamic environment which is more realistic for practical use.

2.4 Autonomous Navigation of Underwater Robot

Basically, autonomous navigation of any robotic vehicle can be interpreted as continuous interaction between perception, intelligence and action [16] as shown in Figure 2.3:

Perception: Various sensing devices are normally employed to extract perceptual knowledge about robot and its surroundings which can be used in further tasks during navigation (such as localization, planning, collision free motion control etc.).

Intelligence: Without any human intervention, robotic vehicle must have learning and inference abilities to adapt required behaviours based on online sensory knowledge about environment.

Action: Based on the acquired knowledge, control signals have been sent to the robot's actuators to perform the predefined actions as per decision taken in intelligence module. Thus navigation of robotic device can be executed autonomously.

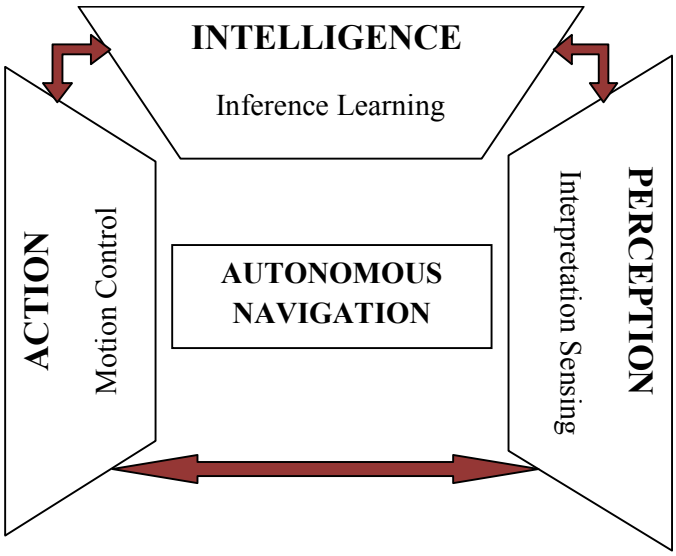


Figure 2.3: Basic Autonomous Navigation Loop

Autonomous navigation from one point to another point between specified start and goal positions can be defined as a group of behaviours. Each behaviour can be considered as a collection of actions or manoeuvres that the UUV performs to achieve a goal.

Intelligent behaviours are on-board capabilities that provide a structure to reason about unexpected or unknown situations or events and allow the UUV to adapt and complete a mission where possible. Intelligent behaviours can be the basis for intelligent autonomy. Intelligent autonomy provides a framework for deliberation and reasoning with intelligent behaviours. The intelligent autonomy framework can use autonomous agents (systems that perceive the environment through sensors and act upon it through actuators) to aid

with intelligent behaviours like decision making, mission planning/re-planning, and fault tolerance. Complex tasks require decision-making abilities rather than programmed choices to autonomously sense situations, make evaluations, and decide on follow-on actions.

The autonomy framework can be classified over a spectrum that includes reactive [26], deliberative / hierarchical [27] and hybrid architectures [28] which can be described as follows:

Reactive architectures use the sense-plan-act schema and are suited to structured and predictable environments and missions. Reactive architectures are subsumptive, "bottom up," sensor-driven, layered and may often be characterized by forward inferencing. Reactive architectures attempt to combine robust subsuming behaviours while avoiding dynamic planning and world models. Reactive architectures appear to behave randomly and achieve success without massive computations by using well-considered behaviours that tend to lead to task completion [29]. Scaling up to complex missions is difficult. Stability and deterministic performance is elusive. Reactive architectures are simpler and reasonably straightforward to implement; however, they do not deal well with deviations from the nominal mission, environment, or health of the UUV.

Hierarchical architectures can be characterized as deliberative, symbolic, structured “top down” approach. They are often implemented using backward inferencing. Hierarchical approaches typically contain world models and use planning and search techniques to achieve strictly defined goals [30]. They take on-board sensor data and process it into information to build and update that world model. The world model is consulted to determine how the AUV could implement the operator’s pre-defined mission goals and ultimately, the commands sent to sensors, motors, control surfaces, and other subsystems. Hierarchical architectures tend to be rigid, unresponsive in unpredicted situations and computation-intensive.

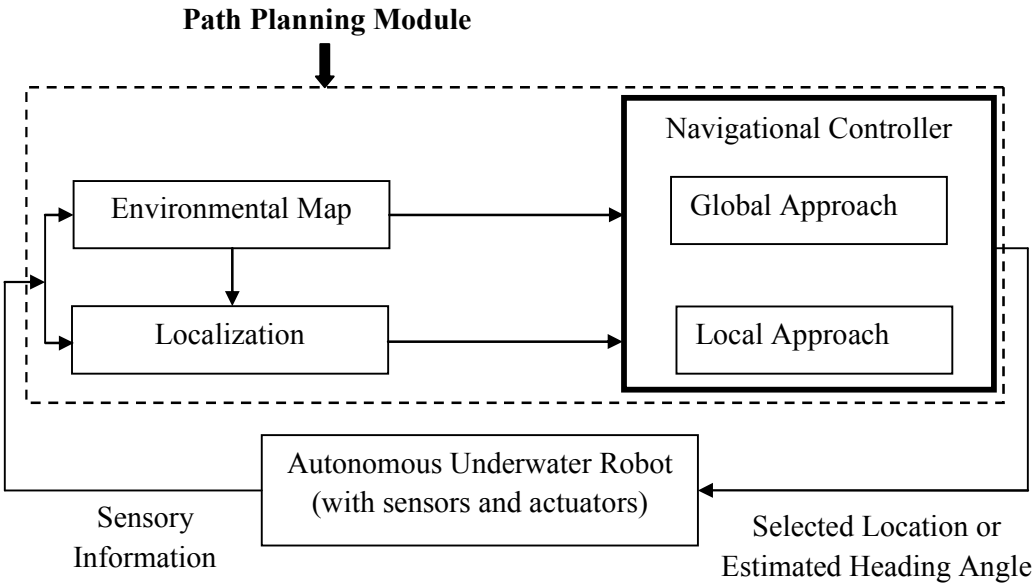


Figure 2.4: Hybrid Approach based Path Planning Framework

Hybrid architectures combine the advantage of fast reactive responses based on a behaviour-based layer coupled with a deliberative layer capable of decisions and planning. In some hybrid architectures, the deliberative layer oversees the sensing and the reactive behaviour-based layer by choosing and prioritizing behaviours. The behaviour-based layer oversees the low-level vehicle controllers and generates commands to the actuators or thrusters as needed. Present research work has concentrated on hybrid approach based navigation to avail benefits from both global and local search approaches. Path planning framework can be seen in Figure 2.4. Enhancement of intelligence in navigational controller is the main objective for development of new path planning algorithm for underwater robot.

2.5 Classical Navigational Strategies for AUV

Adaptive control has been used [12], with the benefits of this type of control obvious due to the changing dynamics of AUVs in the ocean. For example, the controller can adapt itself to vary ocean currents or to a different vehicle density when ballast tanks are used. Adaptive control is also useful because AUVs are usually refitted with new equipment and adapted for different missions which change their static and dynamic characteristics.

Another technique that has been used is sliding mode control [31]. In this control scheme, the dynamics of the system are altered by the application of high-speed switching control.

The system is in essence constrained in such a way so as to exhibit desirable characteristics. This proves useful in the linearization and hence, controlling of underwater vehicle dynamics.

Budiyono [32] has applied coefficient diagram method (CDM) for a robust control design of an autonomous underwater vehicle. The CDM is an algebraic approach in which the characteristic polynomial and the controller are synthesized simultaneously. Particularly, a coefficient diagram (comparable to Bode diagram) is used effectively to convey pertinent design information and as a measure of trade-off between stability, response speed and robustness.

Li and Lee [33] have considered the AUV's diving dynamics as a certain SISO system which possesses the stern plane angle as control input and the depth of the vehicle as output. The two restricting conditions have been broken so that the vehicle could take any pitch angle and the nonlinear dynamics in pitch motion has been assumed to be unknown. Autonomous underwater vehicle motion in ocean conditions requires investigation of new control solutions that guarantee robustness against external parameter uncertainty. A diving-control design based on Lyapunov theory and back-stepping techniques has been proposed and verified by Lapierre [34]. Using adaptive and switching schemes, the control system is able to meet the required robustness. The navigation system should provide a certain estimation of system states within guaranteed error tolerance, and the control scheme must exhibit desirable global performances. Considering the model nonlinearities, the Lyapunov approach has many advantages. A first step allows for designing a control solution that considers the system kinematics and meets global convergence requirements. Then using the backstepping approach [35], the system model is augmented with its dynamic states, while still meeting global performance requirements. Another backstepping stage allows for parameter uncertainty to be taken into account, designing an adaptive scheme that guarantees robustness.

To perform various missions, the designed controller must track the straight line as well as the curved line. Proposed tracking controllers so far did not consider the off-diagonal terms in the system matrix of AUVs due to the difficulty of the controller design [36]. That is, they assumed that the off-diagonal terms in the system matrix are zero. This implies that the shape of bow is the same as that of stern, but the bow and the stern are

not generally symmetric. Thus, the controller using the state transformation was proposed to consider the off-diagonal terms in the system matrix [37].

Because of the uneven distribution of obstacles in the real world, the efficiency of the algorithm decreases if the global environment is represented by regular grids with all of them at the highest resolution. The framed Quadtree data structure has been introduced by Gao et al.[38] for representing the environment efficiently. When planning the path, the dynamic object is expressed instead as several static objects which are used by the path planner to update the path. By taking account of the characteristics of the framed quadtree, objects can be projected on the frame nodes to increase precision of the path.

Recently, Santhakumar and Kim [39] have proposed a new tracking controller for autonomous underwater vehicle-manipulator systems (UVMSs) using the concept of model reference adaptive control. It has also addressed the detailed modelling and simulation of the dynamic coupling between an autonomous underwater vehicle and manipulator system based on Newton–Euler formulation scheme. The proposed adaptation control algorithm has been used to estimate the unknown parameters online and compensate for the rest of the system dynamics. Specifically, the influence of the unknown manipulator mass on the control performance is indirectly captured by means of the adaptive control scheme.

Controller design for AUVs is a difficult and challengeable subject because AUVs are strong nonlinear, large delay, weak communication and observing systems. The performance of the controller depends on the accuracy of the model parameters [40]. So the controller for AUV must have adaptive and robust capability to suppress the uncertain influences from nonlinearity and error of modelling, and the interferences of hydrodynamic interactions from complex external environment. Therefore, the traditional linear control schemes cannot control AUVs accurately after linearizing the model of AUVs for satisfying the need of the control methods.

Repoulias and Papadopoulos [41] have designed a novel closed-loop trajectory tracking controller for an under actuated AUV having 6 degrees of freedom (DOF) and 3 controls, namely a thruster, a rudder and moving surfaces to control the forward, yaw and pitch motions respectively. A backstepping methodology has been adopted as a design tool since it is suitable for the cascaded nature of the vehicle dynamics. It also offers

flexibility and robustness against parametric uncertainties which are often encountered in hydrodynamic modelling.

A motion control strategy consisting of both position and speed control in a horizontal plane has been designed by Wang et al. [42] for different task assignments of underwater vehicles. Combined control of heave and pitch was adopted to compensate for the reduction of vertical tunnel thrusters when the vehicle is moving at a high speed. An improved S-surface controller based on the capacitor plate model was developed with flexible gain selections made possible by different forms of restricting the error and changing the rate of the error.

A numerical study on control effectiveness of a high-speed underwater vehicle with cruciform stern configuration has been performed by Kim and Cho [43] using a computational fluid dynamics approach. The calculation of the control derivatives of the underwater vehicle has been validated by comparison with the experimental results of towing tank tests. A path-following algorithm for diving control of autonomous underwater vehicle (AUV) along a predefined depth has been introduced by He-ming et al. [44] where surge force and stern plane are only available for vehicle's 3DOF diving motion. Controller design is based on robust backstepping technique together with Lyapunov function. The whole system can be guaranteed asymptotic stability with path-following error convergence to zero, and the states are all bounded. A novel active disturbance rejection control (ADRC) controller has been proposed by Yan et al.[45] based on support vector regression (SVR). The SVRADRC has been designed to force an under actuated autonomous underwater vehicle (AUV) to follow a path in the horizontal plane with the ocean current disturbance. It has been derived using SVR algorithm to adjust the coefficients of the nonlinear state error feedback (ELSEF) part in ADRC to deal with nonlinear variations at different operating points.

There are two kinds of classical optimization techniques: direct search method and gradient search method. In the direct search method, only the objective function and constraints are used for the search process, whereas in the gradient search method, the first order and/or second order derivatives are used for the search process. Direct search methods are very slow because many function iterations are required, whereas gradient search methods are faster, but they are inefficient for discontinuous and non-differentiable

functions. Furthermore, both methods seek local optima, thus starting the search in the vicinity of a local optima causes them to miss the global optima.

2.6 AI based Navigation for AUV

Classical control techniques require an exact model of the AUV to be controlled and the solution has been generated by considering the physical phenomena of whole process that govern the process. Although these approaches can be very accurate, they require a lot of information, and the computational cost is very high. Therefore, Artificial Intelligence (AI) based algorithms have been introduced as navigational strategies of underwater robot in past few decades.

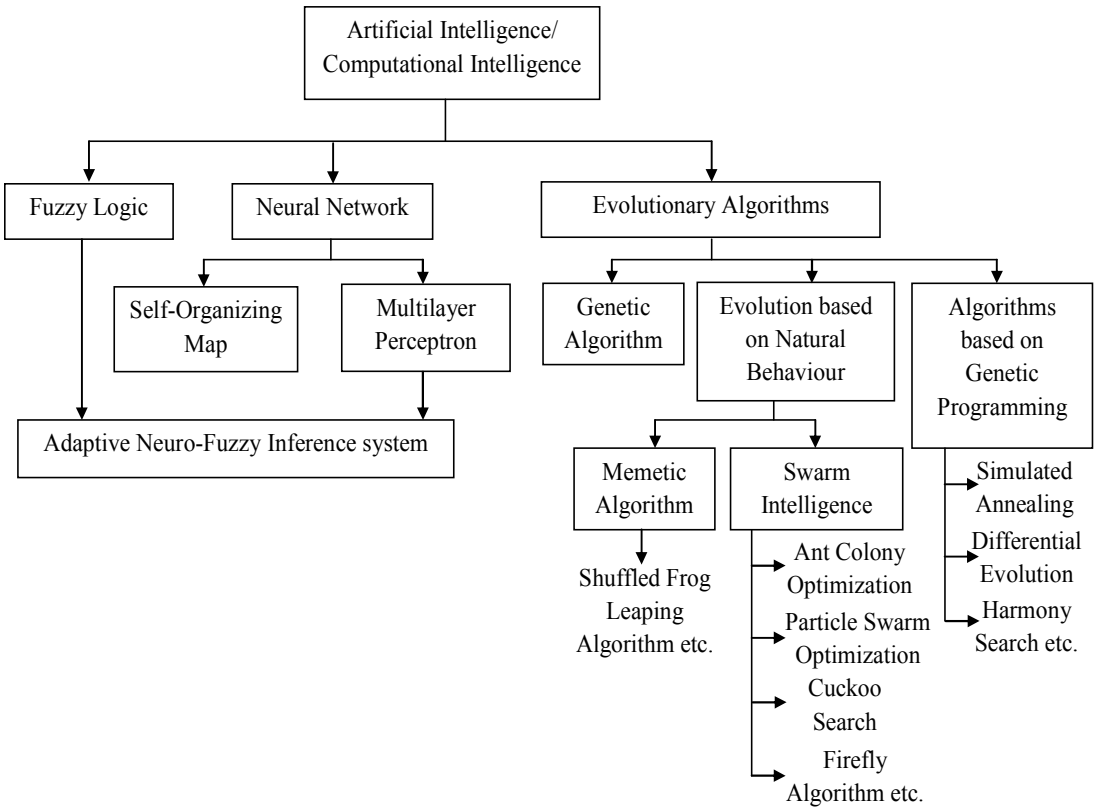


Figure 2.5: Categorization of AI based Computational Algorithms

AI based techniques can be categorised based on their computation mechanisms as shown in Figure 2.5. Among various AI based methods which are mostly used for path optimization purpose have been considered here. In a broad manner, human perception or expertise based decision making algorithms (Fuzzy logic or Neural Network) and evolutionary algorithms are two main constituents of computational intelligence

paradigm. Both sections of Computational Intelligence have the ability to replace the traditional techniques while solving problems of classification, control, prediction, modelling and optimization etc. Fuzzy systems are known for their capabilities to handle ambiguous or vague concepts of human perception for complex systems problems, where it is extremely difficult to describe the system models mathematically. Similarly, NN have the well-known advantages of being able to approximate any nonlinear function and being able to solve problems where the input–output relationship is neither well defined nor easily computable, because NN are data-driven [46]. On the other hand, the evolutionary algorithms have emerged as robust techniques for many complex optimization, identification, learning and adaptation problems [47].

2.6.1 Fuzzy Logic Controller for Navigation

The wide applicability of fuzzy logic in autonomous navigation is mainly based on suitable knowledge representation of inherently vague notions achieved through fuzzy IF-THEN rules [16]. These rules typically contain linguistic information, which describes the problem at hand very simple and fast. Further, in the majority of fuzzy logic application in navigation, a mathematical model of the dynamics of the vehicle is not needed in the design process of the motion controller. Only the problem-specific heuristic control knowledge is needed for the inference engine design. From a more practical point of view, fuzzy logic is the most appropriate modelling tool for representing imprecision and uncertainty of the sensor readings. Another reason that explains the popularity of fuzzy logic in autonomous navigation is the low computation time of the hardware implementations of fuzzy controllers which favours real-time applications. The rationale behind using fuzzy logic is that fuzzy logic has already been proven to be a very useful modelling tool when dealing with problems characterized by the presence of uncertainty [48]; in this case the vehicle's movement and sensing actions depend on a number of environment conditions that are impossible to model. As such, no realistic assumptions can be made about trajectory generation, path planning and collision avoidance. Therefore, sensor based navigation controllers with reactive and reflective capabilities [4, 32] must be derived that generate control commands based on sensor data. Such controllers behave as goal-based controllers when no obstacles are considered or as reaction based controllers when obstacle avoidance is necessary.

Fuzzy logic controllers have been proposed and implemented with success on AUVs in several cases [49, 50, 51] in primitive ages. The nature of fuzzy logic offers a control solution when a mathematical model is not well known, or not known at all as the case may be. Thus, implementing a controller on an AUV using fuzzy logic can avoid the need for complex hydrodynamic modelling of the vehicle. However, the downside is the implementation of the controller itself poses its own level of complexity.

A fuzzy logic based general purpose modular control architecture has been presented by Kanakakis et al. [48] for underwater vehicle autonomous navigation, control and collision avoidance. Three levels of fuzzy controllers comprising the sensor fusion module, the collision avoidance module and the motion control module are derived and implemented. No assumption is made on the specific underwater vehicle type, on the amount of a priori knowledge of the 3-D undersea environment or on static and dynamic obstacle size and velocity. The derived controllers account for vehicle position accuracy and vertical stability in the presence of ocean currents and constraints imposed by the roll motion.

Fuzzy inference system can express the qualitative aspect of human reasoning without using any precise mathematical models of the system [52]. Fuzzy Modelling has been applied by Hassanein et al. [53] to obtain a nonlinear dynamic model of the AUV and also to develop a robust guidance and control system for it. The path planning algorithm based on fuzzy reasoning may suffer from some critical issues: being stuck in local minima and raise in no. of rules to cope up with possible positions of obstacles and targets [54]. The single input fuzzy controller has been employed for underwater navigation by Ishaque et al. [55] by reducing the number of input variables and rules of FLC. Linguistic knowledge based rule base of FLC can be adapted using NN learning mechanism with less computation time and cost [56].

A new hierarchical closed loop fuzzy control system for horizontal plane trajectory tracking of under actuated Autonomous Underwater Vehicles (AUV) has been presented by Raimondi and Melluso [57]. A model for the AUV is formulated introducing a polar coordinates transformation for the velocities in the body fixed frame. It is employed to control the unactuated sway direction, the longitudinal position and the yaw by using the surge force and the yaw torque only. The highest level control is developed by employing a fuzzy inference system for obtaining the guidance control laws. The properties of the fuzzy system ensure forward surge velocity, fast convergence and Lyapunov's stability of

the motion errors. A new low level kinetic controller is presented for ensuring that the actual velocities of the AUV converge to the fuzzy guidance control laws.

A fuzzy behavioural navigation system with a two-layered hierarchical architecture has been proposed by Zhao et al. [58] to address the bottom collision avoidance problem of autonomous underwater vehicles (AUVs) with depth-keeping mission. In the path planning module, depth keeping and collision avoidance behaviours and a fuzzy arbiter are included. The fuzzy logic control has also been used in the pitch motion control module. A novel path planning algorithm based on fuzzy logic for autonomous underwater vehicle (AUV) in 3D unknown space has been proposed by Liu et al. [59], in which the 3D path planning problem is decomposed into two independent planning problems in the horizontal and vertical planes respectively. In each plane, obstacle avoidance and goal seeking behaviours are designed to solve the 2D path planning. The outputs of these two behaviours are fused via weight fuzzy logic controller to generate the final motion command.

Based on autonomous underwater vehicle (AUV), through the analysis of domestic anti-surge technology research status and inrush current model, a control technology with the use of behavioural decision upper and lower piecewise integration control between PID and FPID (Fuzzy PID) has been designed by Tang et al.[60]. Then, according to the characteristics of underwater robotics and inrush current optimal path planning, AUV can get information and plan a different anti-surge behavioural decision, which makes underwater robot better complete the underwater operations.

2.6.2 Neural Network based Navigational Strategy

As neural network has attributes of quick response and approximation for complex nonlinear function [61], huge research interests have spurred for neural network applied underwater motion control problems [62-63]. A neural-network-based learning control scheme for the motion control of autonomous underwater vehicles (AUV) is described by Venugopal et al. [64]. The scheme has a number of advantages over the classical control schemes and conventional adaptive control techniques. The dynamics of the controlled vehicle need not be fully known. The controller with the aid of a gain layer learns the dynamics and adapts fast to give the correct control action. The dynamic response and tracking performance could be accurately controlled by adjusting the network learning

rate. A modified direct control scheme using multi-layered neural network architecture is used in the studies with backpropagation as the learning algorithm.

Parallel neural network, where training speed is enhanced, has been proposed by Liang et al. [65] as AUV motion controller. A linearly parameterized neural network (LPNN) as an adaptive controller has been employed in approximation of AUV dynamics by Li et al. [66]. Back propagation based neural network for navigation and control of AUVs may have issues regarding convergence and stability [67]. Removing all difficulties, a one-layer neural-network (NN) controller with pre-processed input signals have been implemented by Jagannathan and Galan [68] for the approximation of dynamical forces and moments of AUV while tracking a desired trajectory. Prior to designing the controller for AUV, it is believed that the task is challenging due to external disturbance from environment, the unknown nonlinear dynamics of AUVs and high coupling between channels. Therefore neural networks have been used for modelling the nonlinear functions of the system dynamics [69]. As in the most neural networks controllers the neural networks modelling of dynamics plays a significant role, developing a precise model which increases the accuracy of the controller is an important issue. An adaptive neural network control method for spatial path following control of an AUV has been proposed by Zhou et al. [70] to exploit three adaptive RBF based neural network controllers which are based on the Lyapunov stability theorem. As training data for learning mechanism is based on heuristic knowledge, neural network may have slow convergence and lack of generalization due to limited patterns to represent complicated environments [71].

Designing a control law for Autonomous Underwater Vehicle (AUV) has been a considerable challenge from classical and modern control view point. Neural Network based control is seen as an emerging technology for intelligent control of complex system. An approach to model the controller for the AUV using Recurrent Neural Networks (RNN) has been proposed by Babu et al. [72]. RNN had been selected to model the system as it has very good capability to incorporate the dynamics of the system. The AUV dynamic equations had been modelled to obtain the data required for training the neural network. A controller has been developed which can learn change in the dynamics on the fly.

A neural network robust adaptive control algorithm has been employed by Ge et al. [73] in the controller design to overcome the uncertainties of the model and the influence of external disturbances. Meanwhile, the dominant input idea is adopted to reduce the complexity of the closed-loop. The stability performance of the system is proved by using Lyaponov stability theory.

In order to deal with the parameter variations and uncertainties due to time-varying hydrodynamic damps, the radial basis function neural network (RBFNN) has been introduced by Bian et al. [74] to estimate unknown terms where an adaptive law is chosen to guarantee optimal estimation of the weight of NN. Based on the Lyapunov stability theorem, an adaptive NN controller is designed to guarantee all the error states in the path following system are asymptotically stable. In order to deal with the estimation error and current disturbance, a virtual control input has been introduced to ensure that the error system, including position error and heading error which can be converged to zero.

A novel hybrid control approach has been presented by Sun et al. [75] for trajectory tracking control of unmanned underwater vehicles. The kinematic and dynamic controllers are integrated by the proposed control strategy. The paper has two objectives. Firstly, an improved backstep method is proposed to generate the virtual velocity using a bio-inspired neurodynamics model in the kinematic controller. The bio-inspired neurodynamics model is intended to smooth the virtual velocity output to avoid speed jumps of the unmanned underwater vehicle caused by tracking errors and to meet the thruster control constraints. Secondly, a new sliding-mode method is added to the dynamic controller, which is robust against parameter inaccuracy and disturbances. The combined kinematic–dynamic control law is applied to the trajectory tracking problem of two different types of unmanned underwater vehicle.

Park [76] has proposed a neural network (NN)-based tracking control method for under actuated autonomous underwater vehicles (AUVs) with model uncertainties. In order to solve the difficulties in designing the controller for under actuated AUVs, the additional virtual control input has been developed, and the approach angle, which generates the desired yaw angle to track any reference trajectory, is introduced. Moreover, the NNs are used to deal with model uncertainties in the hydrodynamic damping terms of AUVs. Finally, the proposed controller is designed based on the

dynamic surface control (DSC) method, and the boundedness of all tracking errors has been proved by using the Lyapunov stability theory.

2.6.3 Hybrid Fuzzy-Neuro Approach for Navigation

Fuzzy logic's inference ability based on human perception is mingled with universal approximation and learning skills of neural networks based on input-output training pairs in adaptive neuro-fuzzy inference system. In1993, Jang has introduced ANFIS which is nothing but a Takagi-Sugeno fuzzy model trained by BP and LSE in a combined manner [77]. A self-adaptive neuro-fuzzy inference system trained by hybrid parameter learning algorithm has been addressed by Lee et al. [78] in the navigation of autonomous underwater vehicle (AUV). To model the inverse kinematics of the AUV, a self-adaptive learning algorithm for recurrent neural-fuzzy system based on clustering approach has been implemented by Wang et al. [79]. A neural-fuzzy controller based on FMFNN has been applied to the AUV control by Kim and Yuh [80]. For generation of automatic dynamic path in 3D space, a new mathematical approach based on ANFIS has been proposed by Woo and Polisetty [81]. An intelligent control system based on neural-fuzzy controller along with hybrid learning has been implemented by Cherroun and Boumehraz [82] for tracking moving targets by a mobile robot. The capability of adaptive network, fuzzy inference system has been evaluated by Salman et al. [83] for modelling of self-propelled underwater vehicle. It has been observed that the complexity increases with the increase in the number of inputs to ANFIS there is an increase in number of rules exponentially. Adaptive nonlinear output feedback control of GD-FNN (generalized dynamic fuzzy neural network) has been proposed by Chen et al. [84] for controlling the motion of underwater robot. This method is composed of nonlinear and uncertain parts of underwater robot by online adaptive learning algorithm without knowing fuzzy neural structure and training phase in advance. Output feedback control guarantees initial stability of system. Closed loop stability of system has been proved by Lyapunov and structure of controller has found to be simple without accurate mathematical model of system. Numerous researches on adaptive neuro fuzzy inference system based control of autonomous underwater robot have been executed with high success [85-86].

The complexity increases with the increase in the number of inputs to ANFIS there is an increase in number of rules exponentially which results ANFIS model to become complex thus ANFIS has used more number of datasets for training. In effect this can be limitation

of ANFIS model. Further it is observed that by using the ANFIS-EA model there is overall reduction in computational burden. But it appears that ANFIS-EA is more efficient when the number of imprecise parameters is less [87].

2.6.4 Evolutionary Computation based Navigational Strategy

The computational drawbacks of existing derivative-based numerical methods have forced the researchers all over the globe to rely on metaheuristic algorithms founded on simulations to solve engineering optimization problems. A common factor shared by the metaheuristics is that they combine rules and randomness to imitate some natural phenomena. Last few decades have seen an incredible growth in the field of nature-inspired meta-heuristics [88]. Metaheuristic algorithms eradicate some of the afore-mentioned difficulties and are quickly replacing the classical methods in solving practical optimization problems. Metaheuristic algorithms typically intend to find a good solution to an optimization problem by 'trial-and-error' in a reasonable amount of computational time. During the last few decades, several metaheuristic algorithms have been proposed. These algorithms include Genetic Programming, Evolutionary Programming, Evolutionary Strategies, Genetic Algorithms, Differential Evolution, Harmony Search algorithm, Ant Colony Optimization, Particle Swarm Optimization, and firefly Algorithms etc. as depicted in Figure 2.6. Each optimization algorithm uses different properties to keep a balance between the exploration and exploitation goals which can be a key for the success of an algorithm. Exploration attribute of an algorithm enables the algorithm to test several areas in the search space. On the other hand, exploitation attribute makes the algorithm focus the search around the possible candidates [89].

In engineering problems, optimization is to look for a vector that can maximize and minimize a function. Nowadays, stochastic method is generally utilized to cope with optimization problems. Though there are many ways to classify them, a simple one is used to divide them into two groups according to their nature: deterministic and stochastic. Deterministic algorithms can get the same solutions if the initial conditions are unchanged, because they always follow the rigorous move. However, regardless of the initial values, stochastic ones are based on certain stochastic distribution; therefore they generally generate various solutions. In fact, both of them can find satisfactory solutions after some generations. Recently, nature-inspired algorithms are well capable of solving numerical optimization problems more efficiently.

Generally, stochastic algorithms have two types: heuristic and metaheuristic. Recently, nature-inspired metaheuristic algorithms perform powerfully and efficiently in solving modern nonlinear numerical global optimization problems. To some extent, all metaheuristic algorithms strive for making balance between randomization (global search) and local search.

Evolutionary computation has multi-fold advantages, including the simplicity of the approach, its robust response to changing circumstance, its flexibility, and many other facets. The evolutionary algorithm can be applied to problems where heuristic solutions are not available or generally lead to unsatisfactory results. They all share a common conceptual base of simulating the evolution of individual structures via processes of selection, mutation, and reproduction. Compared to other global optimization techniques, evolutionary algorithms (EA) are easy to implement and very often they provide adequate solutions. The flow chart of an EA is illustrated in Figure 2.7. A population of candidate solutions (for the optimization task to be solved) is initialized. New solutions are created by applying reproduction operators (mutation and/or crossover). The fitness (how good the solutions are) of the resulting solutions are evaluated and suitable selection strategy is then applied to determine which solutions are to be maintained into the next generation. The procedure is then iterated.

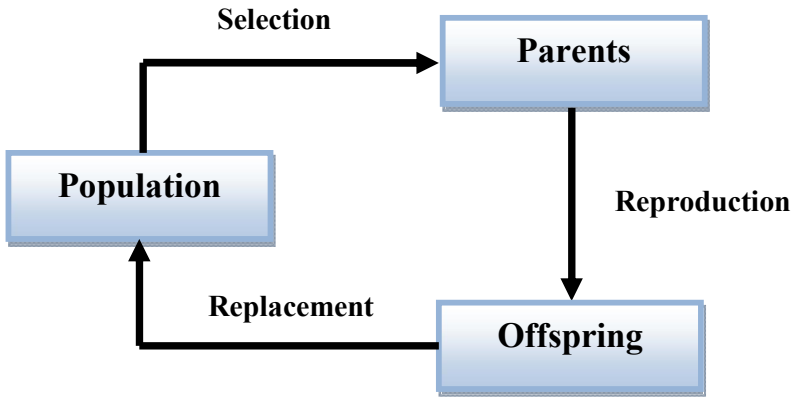


Figure 2.6 Flowchart for Evolutionary algorithms

The population-based collective learning process, self-adaptation, and robustness are some of the key features of evolutionary algorithms when compared to other global optimization techniques. Inappropriate selection of various parameters, representation, etc. is frequently blamed. There is little reason to expect that one can find a uniformly

best algorithm for solving all optimization problems. Evolutionary algorithm behaviour is determined by the exploitation and exploration relationship kept throughout the run. For hybrid evolutionary approaches, the main task is to optimize the performance of the direct evolutionary approach. Recently, hybridization of evolutionary algorithms is getting popular due to their capabilities in handling several real world problems involving complexity, noisy environment, imprecision, uncertainty, and vagueness.

2.6.4.1 Genetic Algorithm

In evolutionary robotics, Genetic algorithms are commonly employed for design of autonomous robotic behaviour controller over years. Sugihara and Yuh [90] have discussed GA-based motion planning in 3D space for underwater robotic vehicles. A genetic algorithm (GA) for path planning can generate a collision-free path in an environment with two kinds of obstacles: Solid and hazardous obstacles such that a path cannot intersect solid obstacles while it can intersect hazardous obstacles at the expense of extra costs. The most important advantage of our GA-based approach is the adaptivity in the sense that a GA can respond to environmental changes (e.g., encounter with an unknown obstacle) and adjust a path globally to the new environment. Therefore, the GA approach can incorporate on-line motion planning with online motion planning.

GA may suffer from imbalance between exploration and exploitation in absence of adaptability in different genetic operators (e.g. crossover, mutation, selection). A GA-inspired AUV path planner associated with dynamic programming has been proposed by Cheng et al. [91]. Taking the advantages of both metaheuristic and exact algorithms, the proposed path planner is more capable of handling large scaled path-planning problem than an exact algorithm-based path planner. Nevertheless, the proposed path planner is able to achieve higher speed and provide solutions with higher quality than its metaheuristic algorithm-based counterparts. An improved cost function has been introduced by Tanakitkorn et al. [92] for a grid-based genetic algorithm (GA) path planning in 2D static environments. The proposed function consists of energy consumption terms that are estimated according to dynamics of a hover-capable AUV - notably Delphin2 AUV. It seeks for a path that requires least effort for the vehicle to move along. Based on the theory and the application of GA, Hong-jian et al. [93] have presented a global path planning method for AUV based on GA and domain knowledge in a large-scale chart. The grid method is adopted to set up a discrete space model for path

planning based on known chart data, and each data structure of a grid stores some property such as digital elevation, permit and so on. A kind of decimal grid-coordinate coding scheme which adopting a variable length chromosome encoding is presented. The generating method of initial population, the fitness evaluation function, the evolve strategy and some superiority genetic operators are all designed and introduced in detail. And some measures are also adopted to improve the searching capability and to speed up convergence of the algorithm. Sun and Zhang [94] have proposed a method of environment modelling based on the environmental information provided by electronic chart which is utilized in the global path planning for AUV, and solved the problem of environment modelling in large-scale marine environment. GA is used as the search strategy of the global path planning, and compared to A* algorithm. In order to resolve the problems such as slow convergence speed etc. of basic genetic algorithm for path planning of AUV, a path planning method for AUV is proposed by Yan et al. [95] based on improved genetic algorithm. This method introduces an improved genetic algorithm and a fitness function with clear physical meaning to increase its search speed and optimization level and resolve path planning problems of Multi-objective optimization for AUV.

Unlike dynamic programming, the computational complexity of soft-computing optimization methods, as GAs, increases linearly with the dimension of the solution space (i.e., logarithmically with the number of nodes). Their drawback is that convergence to the optimal solution is not guaranteed in finite time, so that one may end up with a suboptimal solution.

2.6.4.2 Particle Swarm Intelligence

Swarm Intelligence is an innovative distributed intelligent paradigm for solving optimization problems that originally took its inspiration from the biological examples by swarming, flocking and herding phenomena in vertebrates. Particle swarm optimization algorithm (PSO) is new type swarm intelligence algorithm after genetic algorithm and ant colony optimization algorithm, which is usually used in solving complex problems. Because its iterative formula is continuous, PSO is more suitable to solve route planning without grid. To the problem of premature frequently appeared in standard particle swarm optimization, improved particle swarm optimization (IPSO) algorithm was proposed. IPSO firstly selected elite particles and bad particles according to relevant cost function,

updated velocity of bad particles according to kinetic energy loss of elite particles to avoid premature in search process. Secondly IPSO proposed velocity update strategy with failure experience of worst particles to let particles avoid bad result [96]. Particle swarm optimization (PSO) was also used to solve the problem of 3D path optimization [97].The particle swarm optimization has been applied by Tang et al. [98] to plan path of the underwater vehicle. Firstly, the purpose of dimensionality reduction by the transformation of spatial coordinates and slicing of 3D space is achieved. Secondly, an effective path function and a rotation frequency function are defined. Then the path planning problem is changed into solving the optimization problem, and there are some improvements in the program in order to meet project needs. Path planning method for unmanned underwater vehicles (UUV) homing and docking in movement disorders environment has been proposed by Yan et al. [99]. Firstly, cost function is proposed for path planning. Then, a novel particle swarm optimization (NPSO) is proposed and applied to find the waypoint with minimum value of cost function. A novel improved particle swarm algorithm named competition particle swarm optimization (CPSO) has been proposed by Yan et al. [100] to calibrate the Underwater Transponder coordinates. Zeng et al. [101] have combined a quantum behaved particle swarm optimization (QPSO) algorithm with a cost function which is based on the total time required to travel along the path segments accounting for the effect of space-time variable currents. The proposed path planner is designed to generate an optimal trajectory for an AUV navigating through a spatiotemporal ocean environment in the presence of irregularly shaped terrains as well as obstacles whose position coordinates are uncertain. A completely decentralized three-dimensional topology control algorithm for AUVs has been proposed by Amiri et al. [102]. It is aimed at achieving maximal coverage of the target area. The algorithm enables AUVs to autonomously decide on and adjust their speed and direction based on the information collected from their neighbours. Each AUV selects the best movement at each step by independently executing a Particle Swarm Optimization (PSO) algorithm.

2.6.4.3 Ant Colony Optimization

Ant Colony Algorithm is one of the important methods in dealing with the NP-hard problem. In order to overcome the deficiencies of the algorithm effectively such as slow convergence, easy to fall into local optimal solution, a hybrid adaptive ant colony algorithm has been proposed by Wang et al. [103] and it has been used in the AUV path planning problem. A mathematical model has been established for it and made a practical

engineering example for simulation and analysis of the correlative parameters to the influence on the algorithm.

Global path planning is one of the key techniques of underwater vehicle's intelligent control system, whose purpose is to find a collision-free path from the source position to the destination position according to some optimization criteria. The ant colony algorithm has been used by Liu et al. [104] when studying the global path planning for underwater vehicle in three-dimensional space. The methods for abstract modelling in three-dimensional space were described. In light of the rule of security, economy and the shortest path, the fitness evaluation function was designed while pheromone updating rules, which synthesized iteration-best information and global-best information, were given as well. Path optimization search algorithm based on ACS algorithm was designed by Liu et al. [105], while pheromone representation, route point choosing rules, heuristic functions and pheromone updating rules were given as well.

Global path planning of underwater robot using large-scale chart data has been studied by Wang and Xiong [106] and the principles of ant colony optimization (ACO) were applied. This paper introduced the idea of a visibility graph based on the grid workspace model. It also brought a series of pheromone updating rules for the ACO planning algorithm. The operational steps of the ACO algorithm are proposed as a model for a global path planning method for AUV. To mimic the process of smoothing a planned path, a cutting operator and an insertion-point operator were designed. A path optimization search algorithm based on ACO is proposed by Liu and Dai [107] while pheromone representation, route point choosing rules, heuristic functions and pheromone updating rules are discussed in detail.

2.6.4.4 Shuffled Frog Leaping Algorithm

Over the past decade, many salient methods have been developed to solve these problems. Recently, shuffled frog leaping algorithm (SFLA) which is a memetic metaheuristic for combinatorial optimization was designed to solve water distribution system design problem by Eusuff and Lansey [108]. It has the advantages of simple concept, few parameters, high performance, and easy programming. The SFLA has been tested on several benchmark functions that present its efficiency to many global optimization problems [109]. Its effectiveness and suitability have also been demonstrated for real world complex engineering problems [110-111]. Modified shuffled

frog-leaping algorithm (MSFLA) with new search-acceleration parameter has been introduced for applications to project management in [112] and it was concluded that the MSFLA with an acceleration factor in the range of 1.3–2.1, on average, has the best chance of finding the global optimum with the least number of evolutionary iterations.

An application of SFLA on data clustering is proposed by Amiri et al. [113]. Rahimi-Vahed and Mirzaei [114] have proposed a hybrid version of SFLA and bacteria optimisation algorithms for handling multiple objectives to solve a mixed model assembly line sequencing problem. A novel method of local path optimization based on the position of the target and the obstacles has been addressed. During the process of SFL [115, 116], the position of the globally best frog in each iterative is selected, and reached by the robot in sequence. We assume each obstacle as a point in center of the obstacle and this information is known for our algorithm but the shape, size, and other geometries from the environment and obstacle are unknown and robot path planning processor updates the information on path [117]. With this method the robot can automatically deal with the changes of the unknown environment and get some global optimization effect by using the information of target and obstacles at the same time.

2.6.4.5 Differential Evolution Approach

Easy calibration, stochastic nature and randomization along with fast convergence make DE broadly popular as evolutionary search method in comparison with Genetic Algorithm (GA) [118], Ant Colony Optimization (ACO), Particle Swarm Optimization (PSO), Artificial Immune System (AIS), Artificial Bee Colony (ABC), Cuckoo-Search (CS) etc. [119] for several numerical benchmark problems. The diversity in DE facilitates global path planning where search space for robot is pre-specified. In partially known workspace, robot traces path avoiding near-by obstacles as local path planning scheme which can be achieved through exploitation ability of DE towards optima. Simplified and modified version of GA termed as Differential Evolution embedded with few control parameters can provide automatic balanced search by mainly adapting difference between population vectors [120].

To increase global exploration ability of DE, theory of Chaos has been combined with DE while applying in path planning for mobile robot avoiding obstacles [121-122]. Coordination of multiple robots without hitting each other during navigation has been achieved by Chakraborty et al. [123] by employing a number of DE algorithms in a

parallel manner. Such distributed approach on DE has been found to be effective for quicker convergence in comparison with centralized DE approach or PSO. In a static environment, DE algorithm has been applied by Mo and Meng [124] to globally optimize possible paths for robot as approximated by AVBN modelling. Due to premature convergence, optimal solution of DE may get trapped into local minima situation. For reducing such drawbacks, Bashiri et al. [125] have hybridized two mutation schemes along with adaptively tuned mutation parameters in HADE algorithm which has found success in localization of mobile robots with less number of population members than conventional DE or PSO. 3D collision free path based on B-Spline curve has been planned by Nikolos and Tsourveloudis [126] for both UAVs (Unmanned Aerial Vehicles) and AUVs (Autonomous Underwater Vehicles) considering their respective constraints. Most recently, Zhang and Duan [127] have combined DE with improved version of level comparison technique for planning of UAV's global path in 3D environment. DE based trajectory optimization for underwater glider has been achieved by Zamuda and Sosa [128] and is based on predicted knowledge on sea currents as well as local kinematic model of underwater glider.

2.6.4.6 Harmony Search Algorithm

To improve the performance of original harmony search algorithm a number of variants of harmony search has been discussed in the literature [129].The limitations of constant PAR and bw associated with improvisation stage have been overpowered by IHS algorithm [130] where PAR and bw can vary dynamically with iteration number. Omran and Mahadavi [131] have replaced the term bandwidth by employing global best value of solution vectors in Global-best Harmony Search (GHS) algorithm which is conceptualized by PSO. According to GHS, search process will be completely guided by best solution vectors of better fitness as they have the higher probability to be selected. Based on concept of GHS, Pan et al. [132] has proposed a self-adaptive GHS (SGHS) algorithm by employing a new improvisation scheme and an adaptive learning method for parameters to be varied by value as iteration progresses. Large value of bw favours global search while small value provides fine search towards best solution vector. Local-best Harmony Search algorithm with dynamic sub-populations (DLHS) has also been proposed by Pan et al. [133] to facilitate local search within each sub-HM based on dynamic variation of bw value. Wang and Huang [134] have eliminated rigorous process of selecting HS parameters' values by introducing Self-Adaptive Harmony Search

(SAHS) algorithm. Here, PAR is updated dynamically and based on updated PAR, New Harmony vector will be generated within specified boundaries of decision variable. Novel global harmony search being proposed by Zou et al. [135] is based on genetic mutation as well as novel position updating schemes. To increase explorative power of the population-based harmony search along with exploitative behaviour during selection, Das et al. [136] have proposed an explorative HS (EHS) algorithm where bw is considered to be proportional with the average variation of present set of solution. Yadav et al. [137] have attempted to maintain a proper balance between exploration and exploitation by employing intelligent tuning of HS parameters such as harmony memory size (HMS), Harmony Memory Considering Rate (HMCR) and pitch adjustment strategy. Another New Harmony memory consideration rule based pitch adjustment scheme has been introduced by Chen et al. [138] and is termed as NDHS. Global-best harmony search in an improved version (IGHS) has been initiated by Mohammed El-Abd [139] to change the behavior of the search process from explorative at initial stage to exploitative at final stage. Novel improvisation scheme of IGHS has efficiently combined with already developed parameter updating mechanism. Recently, Xiang et al. [140] have introduced a number of modifications in IGHS algorithm such as opposition based learning scheme for initializing harmony memory with a better quality solution vectors, differential evolution based pitch adjustment for improving local search ability of HS and artificial bee colony algorithm based harmony memory consideration rule to control randomness during global search of GHS. New updating mechanism for HMCR and PAR based on composite function has also been embedded with IGHS.

Geem et al. [141] have effectively implemented harmony search for optimizing multi-objective vehicle routing problem. Jati et al. [142] have enhanced explorative nature of search process by incorporating variable sized chemotactic steps of the bacteria within improved harmony search algorithm which is also successfully implemented in multi-robot path planning. A combination of Quad-tree algorithm and Harmony Search based global optimization has been employed by Panov and Koceska [143] to find out best global path for robot. Global search ability of dynamic harmony search and Lyapunov theory-based local search are combined by Sharma et al. [144] to design fuzzy adaptive controller for vision-based navigation of autonomous mobile robots. Mirkhani et al. [145] have proposed a novel method based on the harmony search (HS) algorithm for robot localization through scan matching. To continue fine tuning process of harmony search

even after optimal value reached, Li and Liu [146] have applied collaboration harmony search in path planning of underwater penetration.

2.6.4.7 Hybridization of Evolutionary Approaches

The different control techniques discussed have more commonly been used in combination with each other. The uniting of these different control techniques brings about the advantage of combining the useful properties of each one to improve the robustness and fault tolerance of the overall controller. Although these heuristic methods do not always guarantee the globally optimal solution, they usually provide a reasonable solution, which is suboptimal but almost the global optimal. But these methods suffer from drawbacks such as large memory requirement, long computation times or premature convergence. Hybrid methods, combining stochastic optimization and deterministic methods are found to be promising in solving complex optimization problems. In these methods, initially stochastic optimization methods are used for search purpose to obtain near optimal solution then deterministic methods are used for fine-tuning that region to get the final solution.

As reported in the literature, there are several types of problems where a direct evolutionary algorithm could fail to obtain a convenient (optimal) solution. Some of the possible reasons for hybridization are as follows [47]:

1. To improve the performance of the evolutionary algorithm (example: speed of convergence)

2. To improve the quality of the solutions obtained by the evolutionary algorithm

3. To incorporate the evolutionary algorithm as part of a larger system

From initialization of population to the generation of offspring, there are lots of opportunities to incorporate other techniques/algorithms etc. Population may be initialized by incorporating known solutions or by using heuristics, local search etc. Local search methods may be incorporated within the initial population members or among the offspring. Evolutionary algorithms may be hybridized by using operators from other algorithms (or algorithms themselves) or by incorporating domain-specific knowledge. Evolutionary algorithm behaviour is determined by the exploitation and exploration relationship kept throughout the run. Adaptive evolutionary algorithms have been built

for inducing exploitation/exploration relationships that avoid the premature convergence problem and optimize the final results. The performances of the evolutionary algorithm can be improved by combining problem-specific knowledge for particular problems.

A fuzzy neural network controller for underwater vehicles has many parameters difficult to tune manually. To reduce the numerous work and subjective uncertainties in manual adjustments, a Hybrid Particle Swarm Optimization (HPSO) algorithm based on immune theory and Nonlinear Decreasing Inertia Weight (NDIW) strategy has been proposed by Zhang et al. [148]. Owing to the restraint factor and NDIW strategy, an HPSO algorithm can effectively prevent premature convergence and keep balance between global and local searching abilities. Meanwhile, the algorithm maintains the ability of handling multimodal and multidimensional problems. The HPSO algorithm has the fastest convergence velocity and finds the best solutions compared to GA, IGA, and basic PSO algorithm in simulation experiments.

2.7 Summary

Study on numerous navigational strategies has revealed that no single technique has been universally accepted. Some techniques work only in ideal conditions and others solve local problems but not global problems. There are many approaches that combine both local and global navigation and seem to be the most effective ones. A large number approaches have been found to capable of generating a valid path or trajectory through a cluttered environment. Classical control and guidance methods have their own disadvantages. The precise control methods need complicated calculation and expensive equipment; they also consume much energy and need much space to be added to AUVs. Thus, they are not suitable for small AUVs. On the other hand, the traditional methods for guidance of AUVs are not adequate because they are often two-dimensional and require complete knowledge about nonlinear modeling of underwater robot. In this context, present research work has been conducted to design and develop new navigational approaches for underwater robot with higher degree of accuracy regarding path optimization and obstacle avoidance. Kinematic and dynamic models of small size underwater robot have been derived to study the practical behavior of underwater robot.

3. Kinematic and Dynamic Modeling of Underwater Robot

To perform underwater tasks and missions autonomously, an underwater robot must be modelled accurately. Underwater robot accomplishing coupled manoeuvers at some speed are known to be highly nonlinear in their dynamics because of the variations of hydrodynamic coefficients which prevent to achieve high degree of autonomy for robot in non-uniform and unstructured underwater environment. Here, a fresh attempt has been made to simplify nonlinear dynamics of underwater robot model which can be employed in shallow water. Stability issues of underwater robot model have been studied along with certain real time assumptions.

3.1 Kinematic Modeling of Underwater Robot

Kinematic model of a vehicle considers the geometrical aspects of motion. Under actuated model of underwater robot imposes the nonholonomic constraints on the linear velocity to restrict certain direction of motion. Therefore, it would be difficult to achieve the complete model of motion. For analysing motion of underwater robot in 6 DOF, it is necessary to work with two right-handed, orthogonal coordinate systems as shown in Figure 3.1. An inertial reference frame is required to measure distances and angles of the body fixed frame during motion [1]. For this purpose, earth-fixed frame has been chosen as reference. For low-speed marine vehicle which has been studied here, the Earth's movement has a negligible effect on the dynamics of the vehicle. Thus, the earth-fixed frame may be considered as an inertial frame, denoted by {E} (o, X_E, Y_E, Z_E). To precisely identify the configuration of a rigid body, the position and orientation of a point on the body with respect to the inertial reference frame must be known. Thus, a reference frame fixed to a chosen point on the body has been denoted by {B} (b, X_B, Y_B, Z_B) in Figure 3.1.

Both the world and body reference frames are right-handed Cartesian coordinate systems. The X_B axis points along the forward direction of the vehicle, defining the vehicle's longitudinal translation. The Y_B axis points through the right hand or starboard side of the vehicle, defining the vehicle's lateral translation. The Z_B axis, which defines the vehicle's vertical translation or depth, is zero at the surface and points downwards; hence, it is positive for increasing depth.

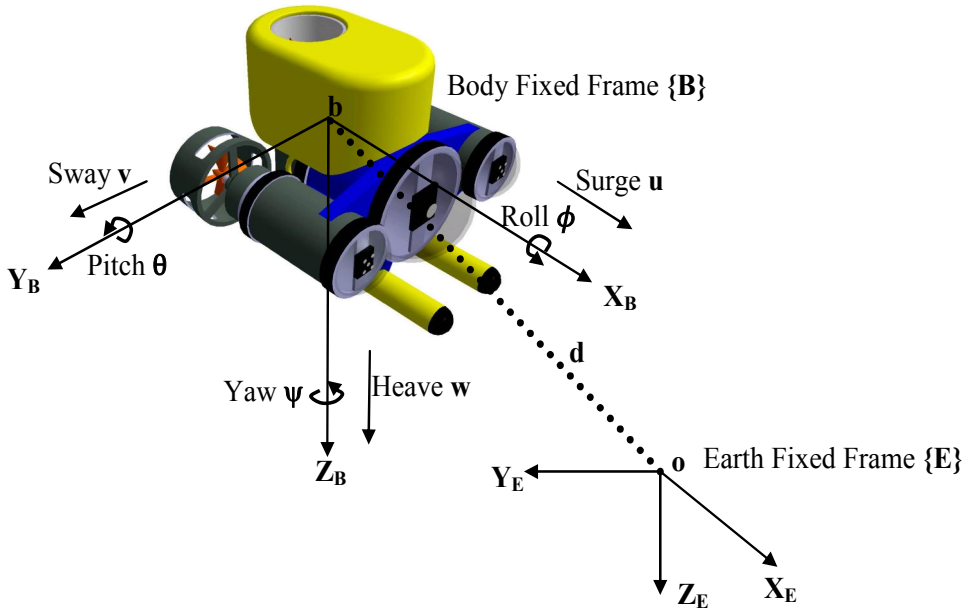


Figure 3.1: Underwater Robot Model with Orthogonal Coordinate Systems

For modeling purpose, it is necessary to compute the underwater robot's position and orientation relative to an earth fixed coordinate system or reference frame {E} [2]. The position and orientation of the vehicle be expressed relative to the spatial frame while the linear and angular velocities are defined relative to the body-fixed frame. Standard notations for these quantities are defined in SNAME (1950) and are reproduced in Table 3.1.

Table 3.1: Standard notations for position and orientation variables are defined in SNAME (1950)

DOF	Motion Definition	Positions and Euler angles (P)	Linear and Angular Velocity (V)	Forces and Moments (λ)
1	SURGE (motion in the x-direction)	x	u	X
2	SWAY (motion in the y-direction)	y	v	Y
3	HEAVE (motion in the z-direction)	z	w	Z
4	ROLL (rotation on x-axis)	ϕ	p	K
5	PITCH (rotation on y-axis)	θ	q	M
6	YAW (rotation on z-axis)	ψ	r	N

The vehicle's position and orientation relative to the earth fixed coordinate system can be represented by a vector, $\eta = \begin{bmatrix} x & y & z & f & \theta & \psi \end{bmatrix}^T = \begin{bmatrix} \eta_1 & \eta_2 \end{bmatrix}^T$, Velocities of the robot represented as, $V = \begin{bmatrix} v_c & \omega \end{bmatrix}^T$ and Forces and moments can be denoted as, $\lambda = \begin{bmatrix} \lambda_1 & \lambda_2 \end{bmatrix}^T$

3.1.1 Linear Velocity Transformation

The translational position and velocity of a vehicle-fixed coordinate frame {B} relative to an inertial coordinate frame {E}, using rotation matrixes. Euler has reasoned that any rotation from one frame to another can be visualised as a sequence of three simple rotations from {E} to {B} [3]. The order of rotations is very important when converting from one coordinate system to another, according to convention (as given in Figure 3.2),

(a) $\{E_1\}$ is the coordinate system {E} after being rotated by a yaw angle of ψ on Z_E axis.

(b) $\{E_2\}$ is the coordinate system $\{E_1\}$ after being rotated by a pitch angle of θ on Y_1 axis.

(c) {B} is the coordinate system $\{E_2\}$ after being rotated by a roll angle of ϕ on X_2 axis.

When converting from world to body coordinates, the reverse order has been used.

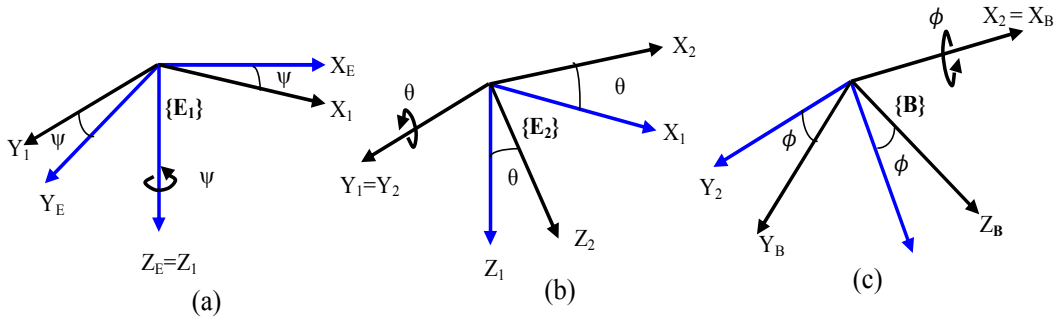


Figure 3.2: Sequence of three rotations through angle ψ, θ, ϕ respectively

By the use of Euler angle representation, three rotation matrices can be defined as follows,

$$R_z(\psi) = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix}, \; R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}, \; R_x(\phi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix}$$

The rotation matrix of frame {B} relative to frame {E} is given as combination of these three matrices and can be illustrated as follows,

$$^{E}R_{B}(\psi,\theta,\phi) = R_{z}(\psi)\cdot R_{y}(\theta)\cdot R_{x}(\phi)$$

$$=>^{E} R_{B}(\psi,\theta,\phi) = \begin{bmatrix} \cos\psi\cos\theta & \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ \sin\psi\cos\theta & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ -\sin\theta & \sin\phi\cos\theta & \cos\phi\cos\theta \end{bmatrix}$$

So, Linear Velocity Transformation can be written as,

$$\dot{\eta}_1 = \begin{bmatrix} \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T = ^{E} R_{B}(\psi,\theta,\phi)\begin{bmatrix} u & v & w \end{bmatrix}^T \tag{3.1}$$

Translational rotation matrix $^{E}R_{B}(\psi,\theta,\phi)$ is orthogonal and hence its inverse will equal the transpose of the matrix, $R^{-1} = R^{T}$. Rotation matrix is pestered by singularity problems at certain angles. In general, an underwater system should be constrained to the following rotation angles, $-\pi < \phi \leq \pi$, $-\pi/2 < \theta < \pi/2$ and $0 \leq \psi < 2\pi$. These are the permitted ranges of Euler angles that do not result in a singular matrix for $^{E}R_{B}(\psi,\theta,\phi)$.

3.1.2 Angular Velocity Transformation

Again the sequence of three simple rotations has to be followed to find out angular velocity of body fixed frame with respect to earth fixed frame as given in Figure 3.2.

1. Due to rotation by an angle of ψ around Z_E axis of coordinate system {E}, {E_1} is the current reference frame whose angular velocity can be expressed as $\begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix}_{E_1}$

2. Due to rotation by an angle of θ around Y_1 axis of coordinate system {E_1}, {E_2} is the current reference frame whose angular velocity can be expressed as $\begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix}_{E_2}$

3. Due to rotation by an angle of ϕ around X_2 axis of coordinate system {E_2}, {B} is the current reference frame whose angular velocity can be expressed as $\begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix}_{B}$

Mathematically, angular velocity of {B} reference frame with respect to {E} frame can be expressed as a sum of all three angular velocities and can be accumulated as follow,

$$\begin{bmatrix} p & q & r \end{bmatrix}^T = \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix}_B + \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix}_{E_2} + \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix}_{E_1} \tag{3.2}$$

In order to do computations involving angular velocities, all these velocities are transformed into body fixed reference frame {B} and sum operation has been done. Rotation matrix $R_x(\phi)$ is used to transform {E_2} reference frame to frame {B}.

$$\begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix}_B = R_x(\phi) \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix}_{E_2}$$

Again, in a same manner, transformation from {E_1} reference frame to frame {B} has been done and can be defined as follows,

$$\begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix}_B = R_x(\phi) R_y(\theta) \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix}_{E_1}$$

Therefore, for angular velocity, the orientation of the body fixed reference frame with respect to the earth fixed reference frame is given by:

$$\begin{bmatrix} p & q & r \end{bmatrix}^T = \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} + R_x(\phi) \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + R_x(\phi) R_y(\theta) \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix}$$

$$= \begin{pmatrix} \dot{\phi} \\ 0 \\ 0 \end{pmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{pmatrix} 0 \\ \dot{\theta} \\ 0 \end{pmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{pmatrix} 0 \\ 0 \\ \dot{\psi} \end{pmatrix}$$

$$= \begin{pmatrix} \dot{\phi} - \sin\theta\dot{\psi} \\ \cos\phi\dot{\theta} + \sin\phi\cos\theta\dot{\psi} \\ -\sin\phi\dot{\theta} + \cos\phi\cos\theta\dot{\psi} \end{pmatrix} = \begin{pmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{pmatrix} \begin{pmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix}$$

$$= {}^E W_B(\phi, \theta).\dot{\eta}_2$$

(3.3)

As ${}^{E}W_{B}(\phi,\theta)$ is undefined or singular for pitch angle of $\theta = \pm 90^{0}$, rotational transformation matrix does not satisfy orthogonal property of matrices. Therefore, transpose of ${}^{E}W_{B}(\phi,\theta)$ cannot be taken as inverse of the matrix.

$$\dot{\eta}_2 = {}^{E}W_{B}(\phi,\theta)^{-1}\begin{bmatrix} p & q & r \end{bmatrix}^{T} = \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi\sec\theta & \cos\phi\sec\theta \end{pmatrix}\begin{pmatrix} p \\ q \\ r \end{pmatrix} \tag{3.4}$$

3.1.3 Kinematic Model in Matrix Form

So, the kinematic equations can be described by two Euler angle representations with different singularities. Summarizing the linear and angular velocity transformation matrices, kinematic equation can generalized for 6 DOF motion of body fixed reference frame {B} with respect to earth fixed reference frame {E} in a compact form. The kinematic model of underwater robot can be depicted as follows [3]:

$$\begin{pmatrix} \dot{\eta}_1 \\ \dot{\eta}_2 \end{pmatrix} = \begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \\ \dot{f} \\ \dot{\theta} \\ \dot{\psi} \end{pmatrix} = \begin{pmatrix} {}^{E}R_{B}(\psi,\theta,f) & 0_{3\times 3} \\ 0_{3\times 3} & {}^{E}W_{B}(\phi,\theta)^{-1} \end{pmatrix}\begin{pmatrix} u \\ v \\ w \\ p \\ q \\ r \end{pmatrix} \tag{3.5}$$

$$\Rightarrow \dot{\eta} = J(\eta_2) \bullet V$$

3.2 Dynamic Model for Underwater Robot

The dynamics of underwater robot contains highly nonlinear and coupled terms which make the mathematical modelling difficult. The forces and moments of a simple underwater robot in three dimensions have been presented in Table 3.1.

For underwater robots, it is desirable to derive the equations of motion for an arbitrary origin in a local body-fixed system to take advantages of the vehicle's geometrical properties. The dynamic behaviour can be described through Newton's laws of linear and angular momentum [149]. Newton's second law relates the force exerted on a body to the lineal acceleration provoked. Considering Newton's second laws in terms of conservation of both linear and angular momentum, the following expression can be written as follows:

Linear Momentum, $p = mv$ and Force acting on body due to translational motion, $F \triangleq \dot{p}$

Angular Momentum, $h = I\omega = p \times r$ and Moment acting on body due to rotational motion, $\tau \triangleq \dot{h}$

By formulating Newton's laws in a body-fixed coordinate system, hydrodynamic and kinematic forces and moments are pretended to be constant due to changes of the vehicle's orientation relative to the global earth fixed reference frame. Assumptions for deriving the equations of motion:

(a) The vehicle is rigid, i.e. no forces acting between individual elements of mass.

(b) The earth is fixed in space, i.e. no forces due to earth's motion relative a star-fixed reference frame.

(c) It will also be assumed that mass is constant in time ($\dot{m} = 0$).

Suppose, B is an arbitrary point in the body where body fixed frame has been assumed and a small piece of object labelled $\boldsymbol{m_i}$ at a distance $\boldsymbol{r_i}$ from the origin, with moment arm $\mathbf{r_p}$ perpendicular to rotation axis (Figure 3.3). Velocity of small element $\boldsymbol{v_i}$ is also perpendicular to both r_i and angular velocity, ω.

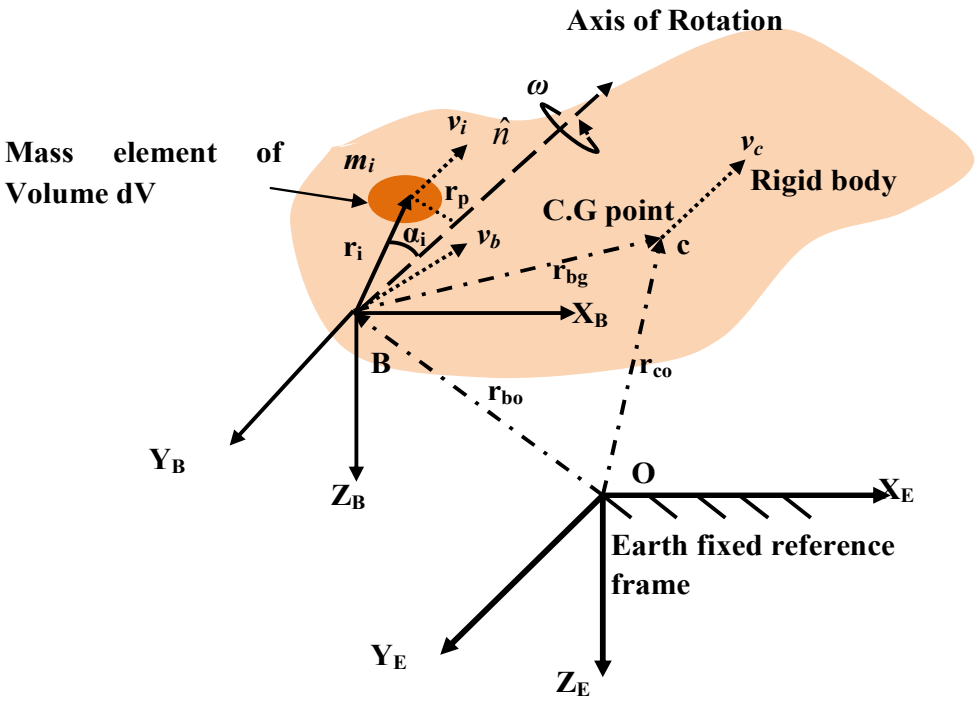


Figure 3.3: An infinitesimal mass element within rigid body of Underwater Robot

The scalar moment of inertia of the object about the rotation axis is as follows,

$$I = \sum m_i r_p^{\,2} \text{ and } r_p = \left| r_i \sin \alpha_i \right| = \left| \vec{r}_i \times \hat{n} \right| \tag{3.6}$$

where, α_i is the angle between the axis of rotation $\hat{n}$ (unit vector) and the radius vector $\boldsymbol{r_i}$.

To express initial inertia in terms of the Cartesian coordinate's (X, Y, Z) and the direction cosines, the position of the infinitesimal mass is considered as, $\vec{r}_i = x_i\hat{i} + y_i\hat{j} + z_i\hat{k}$. The unit vector can be written in terms of the Cartesian unit vectors and the direction cosines, $\hat{n} = \hat{i}\cos\alpha + \hat{j}\cos\beta + \hat{k}\cos\gamma$. After computing $\left|\vec{r}_i \times \hat{n}\right|$, the moment of inertia relative to the axis of rotation can be derived as follows:

$$\begin{aligned} I = {} & \sum m_i (y_i^{\,2} + z_i^{\,2}) \cos^2\alpha + \sum m_i (x_i^{\,2} + z_i^{\,2}) \cos^2\beta + \sum m_i (x_i^{\,2} + y_i^{\,2}) \cos^2\gamma \\ & -2\sum m_i y_i z_i \cos\beta\cos\gamma - 2\sum m_i x_i z_i \cos\alpha\cos\gamma - 2\sum m_i x_i y_i \cos\alpha\cos\beta \end{aligned} \tag{3.7}$$

Denoting the terms of moment of inertia about the coordinate axes and also product of inertia with modified notations, we can express the equation as follows,

$$I = I_{xx}\cos^2\alpha + I_{yy}\cos^2\beta + I_{zz}\cos^2\gamma + 2I_{yz}\cos\beta\cos\gamma + 2I_{xz}\cos\alpha\cos\gamma + I_{xy}\cos\alpha\cos\beta$$

Similarly, unit vector $\hat{n}$ can be represented as column vector, $\hat{n} = \begin{pmatrix} \cos\alpha & \cos\beta & \cos\gamma \end{pmatrix}^T$

The moment of inertia tensor in a matrix form, $\vec{I} = \begin{pmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{pmatrix}$ Where, $I_{yz} = I_{zy}$,

$I_{yx} = I_{xy}$ and $I_{xz} = I_{zx}$.

The scalar moment of inertia can be minimized in a simple matrix multiplication form, $I = \hat{n}^T \cdot \vec{I} \cdot \hat{n}$ (3.8)

The study of dynamics derives the equations of motion for an arbitrary origin in a local body-fixed rotating coordinated system. Suppose, c is an arbitrary vector in both frame, $\dot{c}$ is time derivative of c in earth-fixed frame and c' is time derivative of c in body-fixed frame.

$$\dot{c} = c' + \omega \times c \tag{3.9}$$

This relates the time derivatives of one vector in two different reference frames. Substituting ω for c in this relation shows that angular acceleration is invariant under change of frame.

3.2.1 Translational Motion

To determine the position of C.G with respect to inertial earth-fixed frame, following relation can be found from given Figure 3.3, $r_{co} = r_{bo} + r_{bg}$ (3.10)

Hence, velocity of centre of gravity can be expressed as, $v_c = \dot{r}_{co} = \dot{r}_{bo} + \dot{r}_{bg}$ (3.11)

According to definition of velocity, $\dot{r}_{bg} = r'_{bg} + \omega \times r_{bg}$ (3.12)

Assumption of constant mass for rigid body robot implies that origin of body-fixed reference frame is fixed with respect to centre of gravity, i.e., $r'_{bg} = 0$.

So, $\dot{r}_{bg} = \omega \times r_{bg}$ (3.13)

Considering the velocity of origin of body fixed frame as, $v_b = \dot{r}_{bo}$, the velocity of centre of gravity can be computed as, $v_c = v_b + \omega \times r_{bg}$ (3.14)

To derive equations relating forces and linear momentum, acceleration vector can be found by taking derivative of above equation,

$$\begin{aligned} &\dot{v}_c = \dot{v}_b + \dot{\omega} \times r_{bg} + \omega \times \dot{r}_{bg} \\ &\Rightarrow \dot{v}_c = v'_b + \omega \times v_b + \dot{\omega} \times r_{bg} + \omega \times (\omega \times r_{bg}) \end{aligned} \quad (3.15)$$

Substituting this in equation for forces acting on body-fixed frame, finally yields,

$$m(v'_b + \omega \times v_b + \dot{\omega} \times r_{bg} + \omega \times (\omega \times r_{bg})) = F \quad (3.16)$$

If the origin of body-fixed reference frame is chosen to coincide with the vehicle's centre of gravity, i.e. $r_{bg} = 0$ and $v_b = v_c$,

So the above equation will be minimised as $m(v'_c + \omega \times v_c) = F$ (3.17)

3.2.2 Rotational Motion

Similar approach can be used to obtain rotational equations of motion. The absolute angular momentum about B, origin of body-fixed frame can be defined as, $h = mv_c \times r_i$

Differentiating the above expression with respect to time confers,

$$\dot{h} = m(\dot{v}_c \times r_i) + m(v_c \times \dot{r}_i) \tag{3.18}$$

The first term in the right hand side of the above expression, is the moment vector τ_0 and we can express the velocity of the rigid body as the sum of the velocity of O, origin earth fixed inertial frame as seen in the body-fixed frame and the velocity of a differential volume element with respect to origin B,

$$\begin{aligned} & v_c = \dot{r}_i + \dot{r}_{bo} \\ & \Rightarrow \dot{r}_i = v_c - v_b \end{aligned} \tag{3.19}$$

By using this relation, we can rewrite equation (3.18),

$$\begin{aligned} & \dot{h} = \tau_0 + m(v_c \times (v_c - v_b)) \\ & \Rightarrow \dot{h} = \tau_0 - m v_c \times v_b \end{aligned} \tag{3.20}$$

Substituting eq. 3.14 in the above equation,

$$\dot{h} = \tau_0 - m(\omega \times r_{bg}) \times v_b \tag{3.21}$$

In another approach, equation for absolute angular momentum can be rewritten as,

$$h = m v_c \times r_i = m(v_b + \omega \times r_{bg}) \times r_i = m v_b \times r_i + m r_i \times (\omega \times r_{bg}) \tag{3.22}$$

In the above equation, last term of right hand side can be denoted as inertia tensor or angular momentum for body frame, $m r_i \times (\omega \times r_{bg}) = I\omega$ as the relation, $v_c - v_b = \omega \times r_{bg} = \dot{r}_i = v_i$ has been used here.

So, the expression for angular momentum reduces as, $h = m v_b \times r_i + I\omega$ (3.23)

Differentiating the above expression with respect to time and considering inertia tensor is constant with variation in time,

$$\dot{h} = I\dot{\omega} + m v_b \times \dot{r}_i + m \dot{v}_b \times r_i = I\omega' + \omega \times (I\omega) + m v_b \times (\omega \times r_{bg}) + m(v_b' + \omega \times v_b) \times r_i \tag{3.24}$$

For eliminating $\dot{h}$, equating the equations (3.21) and (3.24),

$$\tau_0 - m(\omega \times r_{bg}) \times v_b = I\omega' + \omega \times (I\omega) + m v_b \times (\omega \times r_{bg}) + m(v_b' + \omega \times v_b) \times r_i \tag{3.25}$$

Using the relation, $v_b \times (\omega \times r_{bg}) = -(\omega \times r_{bg}) \times v_b$, the equation for rotational motion will be reduced as, $\tau_0 = I\omega' + \omega \times (I\omega) + m(v_b' + \omega \times v_b) \times r_i$ (3.26)

If the origin of the body-fixed coordinate system is assumed to be coincide with the centre of gravity of robot's body, then equation will be reduced as,

$$\tau_0 = I\omega' + \omega \times (I\omega) \tag{3.27}$$

The rotational equations for motion are referred as Euler equations.

3.2.3 6-DOF Rigid Body Equations of Motion

Applying Newtonian formalism in rigid body dynamics, general equations of motion for a six degree-of-freedom rigid body is expressed in a chosen local coordinate system are given according to SNAME (1950) notations:

$F = m_1 = \begin{bmatrix} X & Y & Z \end{bmatrix}^T$; External forces

$\tau = m_2 = \begin{bmatrix} K & M & N \end{bmatrix}^T$; Moment of external forces about B

$v_c = V_1 = \begin{bmatrix} u & v & w \end{bmatrix}^T$; Linear velocity of Body-fixed frame

$\omega = V_2 = \begin{bmatrix} p & q & r \end{bmatrix}^T$; Angular velocity of Body-fixed frame

$r_{bg} = \begin{bmatrix} x_g & y_g & z_g \end{bmatrix}^T$; Centre of gravity with respect to body fixed frame

Applying these notations in equation (3.16) and (3.26):

$$\left.\begin{aligned}
&m[\dot{u} - vr + wq - x_g(q^2 + r^2) + y_g(pq - \dot{r}) + z_g(pr + \dot{q})] = X \\
&m[\dot{v} - wp + ur - y_g(r^2 + p^2) + z_g(qr - \dot{p}) + x_g(qp + \dot{r})] = Y \\
&m[\dot{w} - uq + vp - z_g(p^2 + q^2) + x_g(rp - \dot{q}) + y_g(rq + \dot{p})] = Z \\
&I_x\dot{p} + (I_z - I_y)qr - (\dot{r} + pq)I_{xz} + (r^2 - q^2)I_{yz} + (pr - \dot{q})I_{xy} + m[y_g(\dot{w} - uq + vp) - z_g(\dot{v} - wp + ur)] = K \\
&I_y\dot{q} + (I_x - I_z)rp - (\dot{p} + qr)I_{xy} + (p^2 - r^2)I_{zx} + (qp - \dot{r})I_{yz} + m[z_g(\dot{u} - vr + wq) - x_g(\dot{w} - uq + vp)] = M \\
&I_z\dot{r} + (I_y - I_x)pq - (\dot{q} + rp)I_{yz} + (q^2 - p^2)I_{xy} + (rq - \dot{p})I_{zx} + m[x_g(\dot{v} - wp + ur) - y_g(\dot{u} - vr + wq)] = N
\end{aligned}\right\} \tag{3.28}$$

The above six equations are mostly used for representing control and guidance architecture of three-dimensional motion as per Newton's laws i.e. Newton's laws formulated in a body-fixed coordinate system with an arbitrary origin and constant mass and inertia tensor. The first three equations represent the translational motion while last three one represents the rotational motion, totally 6 degrees of freedom. It is desirable to select the axes of the local coordinate system to be the principal axes of inertia of vehicle.

This simplifies the above equations as the terms including products of inertia become zero. This automatically satisfies if the XY, XZ and YZ planes are plane of symmetry.

The dynamic model presented in this section is based on the underwater robotic models proposed by Fossen [149] and Yuh [2]. The dynamic model of underwater robot has been derived from the Newton-Euler motion equation as given follows,

$$M\dot{V} + C(V)V + D(V)V + G = \tau \tag{3.29}$$

where, M is a mass and inertia matrix, C(V) is a Coriolis and centripetal matrix, D(v) is a hydrodynamic damping matrix, G is the gravitational and buoyancy vector, τ is the external force and torque input vector, and v is the velocity state vector. Environmental forces have not been considered in eq. 3.29.

M consists of both a rigid body mass and inertia, M_{RB}, and a hydrodynamic added mass, M_A as follows [3], $M = M_{RB} + M_A$ (3.30)

If frame {B} is positioned at the vehicle's center of gravity (CG), then M_{RB} is expressed as,

$$M_{RB} = \begin{bmatrix} m & 0 & 0 & 0 & 0 & 0 \\ 0 & m & 0 & 0 & 0 & 0 \\ 0 & 0 & m & 0 & 0 & 0 \\ 0 & 0 & 0 & I_x & -I_{xy} & -I_{xz} \\ 0 & 0 & 0 & -I_{yx} & I_y & -I_{yz} \\ 0 & 0 & 0 & -I_{zx} & -I_{zy} & I_z \end{bmatrix}$$

where, 'm' is the mass of the vehicle while the term 'I' represent the inertial tensors. The parameters of the added mass matrix are dependent on the shape of the vehicle; however, they are constants when the vehicle is fully submerged. The parameters are usually in the vicinity of 10% to 100% of the corresponding parameters in the rigid body mass matrix. The effect of added mass can be modelled by a matrix as follows,

$$M_A = \begin{bmatrix} X_{\dot{u}} & X_{\dot{v}} & X_{\dot{w}} & X_{\dot{p}} & X_{\dot{q}} & X_{\dot{r}} \\ Y_{\dot{u}} & Y_{\dot{v}} & Y_{\dot{w}} & Y_{\dot{p}} & Y_{\dot{q}} & Y_{\dot{r}} \\ Z_{\dot{u}} & Z_{\dot{v}} & Z_{\dot{w}} & Z_{\dot{p}} & Z_{\dot{q}} & Z_{\dot{r}} \\ K_{\dot{u}} & K_{\dot{v}} & K_{\dot{w}} & K_{\dot{p}} & K_{\dot{q}} & K_{\dot{r}} \\ M_{\dot{u}} & M_{\dot{v}} & M_{\dot{w}} & M_{\dot{p}} & M_{\dot{q}} & M_{\dot{r}} \\ N_{\dot{u}} & N_{\dot{v}} & N_{\dot{w}} & N_{\dot{p}} & N_{\dot{q}} & N_{\dot{r}} \end{bmatrix}$$

C (V), as with the mass matrix consists of two matrices:

$$C\ (V) = C_{RB}(V) + C_A(V) \tag{3.31}$$

$C_{RB}(V)$ is the rigid body Coriolis and centripetal matrix induced by M_{RB}, while C_A (V) is a Coriolis-like matrix induced by M_A.

If frame {B} is positioned at the centre of gravity of the vehicle, then C_{RB} can be expressed as,

$$C_{RB}(v) = \begin{bmatrix} 0 & 0 & 0 & 0 & mw & -mv \\ 0 & 0 & 0 & -mw & 0 & mu \\ 0 & 0 & 0 & mv & -mu & 0 \\ 0 & -mw & mv & 0 & -I_{yz}q - I_{xz}p + I_z r & I_{yz}r + I_{xy}p - I_y q \\ mw & 0 & -mu & I_{yz}q + I_{xz}p - I_z r & 0 & -I_{xz}r - I_{xy}q + I_x p \\ -mv & mu & 0 & -I_{yz}r - I_{xy}p + I_y q & I_{xz}r + I_{xy}q - I_x p & 0 \end{bmatrix}$$

The Coriolis-like matrix, C_A, is expressed as,

$$C_A(V) = \begin{bmatrix} 0 & 0 & 0 & 0 & -a3 & a2 \\ 0 & 0 & 0 & a3 & 0 & -a1 \\ 0 & 0 & 0 & -a2 & 0a1 & 0 \\ 0 & -a3 & a2 & 0 & -b3 & b2 \\ a3 & 0 & -a1 & b3 & 0 & -b1 \\ -a2 & a1 & 0 & -b2 & b1 & 0 \end{bmatrix}$$

Where,

$$a1 = X_{\dot{u}}u + X_{\dot{v}}v + X_{\dot{w}}w + X_{\dot{p}}p + X_{\dot{q}}q + X_{\dot{r}}r$$
$$a2 = X_{\dot{v}}u + Y_{\dot{v}}v + Y_{\dot{w}}w + Y_{\dot{p}}p + Y_{\dot{q}}q + Y_{\dot{r}}r$$
$$a3 = X_{\dot{w}}u + Y_{\dot{w}}v + Z_{\dot{w}}w + Z_{\dot{p}}p + Z_{\dot{q}}q + Z_{\dot{r}}r$$
$$b1 = X_{\dot{p}}u + Y_{\dot{p}}v + Z_{\dot{p}}w + K_{\dot{p}}p + K_{\dot{q}}q + K_{\dot{r}}r$$
$$b2 = X_{\dot{q}}u + Y_{\dot{q}}v + Z_{\dot{q}}w + K_{\dot{q}}p + M_{\dot{q}}q + M_{\dot{r}}r$$
$$b3 = X_{\dot{r}}u + Y_{\dot{r}}v + Z_{\dot{r}}w + K_{\dot{r}}p + M_{\dot{r}}q + N_{\dot{r}}r$$

The hydrodynamic damping matrix represents the drag and lift forces acting on a moving underwater vehicle. However, for a low-speed underwater vehicle, the lift forces can be considered negligible when compared to the drag forces. These drag forces can be separated into two different terms consisting of a linear and quadratic term [74] as follows, $D(v) = diag\{D_L + D_Q|v|\}$ (3.32)

where D_L and D_Q are the linear and quadratic damping terms respectively and can be represented as, $D_L = diag\{X_u \quad Y_v \quad Z_w \quad K_p \quad M_q \quad N_r\}$ and $D_Q = diag\{X_{u|u|} \quad Y_{v|v|} \quad Z_{w|w|} \quad K_{p|p|} \quad M_{q|q|} \quad N_{r|r|}\}$

The gravitational and buoyancy vector, G, is defined as, $G = \begin{bmatrix} f_B + f_G \\ r_B \times f_B + r_G \times f_G \end{bmatrix}$ (3.33)

Where, f_B is the buoyant force vector, defined as, $f_B = R^{-1}\begin{bmatrix} 0 \\ 0 \\ -B \end{bmatrix}$

and f_G is the gravitational force vector defined as, $f_G = R^{-1}\begin{bmatrix} W \\ 0 \\ 0 \end{bmatrix}$

while r_B is the centre of buoyancy and r_G is the centre of gravity or mass in frame *{B}*. Now, because frame *{B}* is positioned at the centre of gravity, then r_G = [0 0 0]^T, and hence G simplifies to, $\begin{bmatrix} f_B + f_G \\ r_B \times f_B \end{bmatrix}$

Defining r_B as $\begin{bmatrix} x_B & y_B & z_B \end{bmatrix}^T$, that is, the distance in x, y, z coordinates from the origin of frame *{B}*, and by combining above equations,

$$G = \begin{bmatrix} (B-W)\sin\theta \\ -(B-W)\sin\phi\cos\theta \\ -(B-W)\cos\phi\cos\theta \\ B\cos\theta(z_B\sin\phi - y_B\cos\phi) \\ B(x_B\cos\phi\cos\theta + z_B\sin\theta) \\ -B(x_B\sin\phi\cos\theta + y_B\sin\theta) \end{bmatrix} \tag{3.34}$$

The external force and torque vector produced by the thrusters is defined as,

$$\tau = LU \tag{3.35}$$

where L is a mapping matrix and U is a thrust vector.

U is the vector of thrusts produced by the vehicle's thrusters, $U = \begin{bmatrix} \tau_1 \\ \tau_2 \\ . \\ . \\ . \\ \tau_n \end{bmatrix}$

The number of thrust values in U depends on the number of thrusters on the vehicle. The mapping matrix L is essentially a $6 \times n$ matrix that uses U to find the overall forces and moments acting on the vehicle.

3.3 Simplified Model of Underwater Robot

In present research work, a small size underwater robot (Figure 3.1) has been actuated for surge, heave and yaw motions within 3-D workspace of calm water. Major portion of robot's weight has been placed at bottom side vehicle, so that, roll and pitch motion will be automatically controlled. Therefore, underwater robot will always maintain in a horizontal posture which may simplify the nonlinear mathematical model of underwater robot [5]. Non-holonomic constraints on six DOF motion of underwater robot induce high degree of nonlinearity in vehicle's dynamics. As specified underwater robot has very small spinning radius compared to its huge workspace, so, the influence of non-holonomic constraints on motion may be ignored. The speed of the robot relative to the water flow is also assumed to be constant. The required drag force for considered elliptical shape underwater robot (Figure 3.1) is found to be less than cylindrical shape AUVs.

Due to asymmetric body structure of the AUV, dynamic modelling, for sake of convenience, the following assumptions has been made.

1) Centre of mass (CM) and centre of buoyancy (CB) coincide with each other.

2) Mass distribution all over the body is homogeneous.

3) The hydrodynamic terms of higher order as well as pitch and role motions are negligible.

With the assumption of no motion in roll and pitch directions, the vehicle's position and orientation with respect to the earth fixed frame can be reduced to $P_B = \begin{bmatrix} x & y & z & \psi \end{bmatrix}^T$ and kinematic model of AUV (eq. 3.5) can also be simplified as follows [150]:

$$\left.\begin{aligned} \dot{x} &= uCos\psi - vSin\psi \\ \dot{y} &= uSin\psi + vCos\psi \\ \dot{z} &= w \\ \dot{\psi} &= r \end{aligned}\right\} \qquad (3.36)$$

For a neutrally buoyant underwater vehicle, both the CG (Centre of Gravity) and the CB (Centre of Buoyancy) must be located on vertical axis due to its symmetric structure [3]. Conventionally, damping factor for high speed motions must be highly nonlinear and coupled with each other. As the small size robot is assumed of low speed compared to other AUVs, so, only non-coupled linear and quadratic damping terms of diagonal section in damping matrix can be considered for analysis. So, the dynamic equations of underwater motion (eq. 3.28) without roll and pitch motions can be illustrated in a reduced form [150]:

$$\left.\begin{aligned} m\dot{u} - mvr + X_u u + X_{|u|u}|u|u &= \tau_u \\ m\dot{v} + mur + Y_v v + Y_{|v|v}|v|v &= 0 \\ m\dot{w} + X_w w + X_{|w|w}|w|w &= \tau_w \\ I_z\dot{r} + N_r r + N_{|r|r}|r|r &= \tau_r \end{aligned}\right\} \qquad (3.37)$$

Where, τ_u, τ_w and τ_r : Control forces or torques along the surge, heave and yaw motions of robot respectively and $X_u u$, $X_{|u|u}|u|u$, $Y_v v$, $Y_{|v|v}|v|v$, $X_w w$, $X_{|w|w}|w|w$, $N_r r$ and $N_{|r|r}|r|r$: Linear and quadratic drag coefficients.

The derived control configuration in eq. 3.37 can represent the dynamics of small size under actuated underwater robot with three thrusters: two for horizontal and one for vertical motion.

3.4 Stability Analysis based on Lyapunov Function

Stability of underwater vehicle can be defined as the ability of returning to an equilibrium state of motion after a disturbance without any corrective action, such as use of thruster power or control surfaces. Manoeuvrability can be defined as the capability of the vehicle to carry out specific manoeuvres. Excessive stability implies that the control effort will be excessive while a marginally stable vehicle is easy to control. Thus a compromise between stability and manoeuvrability must be made. Furthermore, it makes to distinguish between controls-fixed (open-loop) and controls-free (closed-loop) stability. The essential difference between these terms can be stated as follows [1]:

- ✓ Open-loop stability implies investigating the vehicle's stability when the control surfaces are fixed, and when the thrust from all the thrusters is constant.
- ✓ Closed-loop stability refers to the case when both the control surfaces and the thruster power are allowed to vary. This implies that the dynamics of the control system must also be considered in the stability analysis.

Present research work has considered open-loop stability analysis as forces exerted by thrusters of underwater robot have been assumed to be constant. In control fixed analysis, static stability criteria have been found out based on the hydrodynamic derivatives. For a linear model, stability can be analysed using Routh and Hurwitz criterion. As dynamic model of underwater robot has been considered as nonlinear one, so, Lyapunov's direct method has been implemented here for stability analysis. Lyapunov's direct method uses an energy-like function called Lyapunpov function to study the behaviours of dynamical system analytically. This function can reflect the physical properties of the system under study. Assume the Lypunov candidate function,

$$V(\eta,\dot{\eta}) = \frac{1}{2}\dot{\eta}^T M(\eta)\dot{\eta} + \int_0^{\eta} g^T(z)dz \tag{3.39}$$

Here, V can be interpreted as the sum of kinetic and potential energy of the vehicle. Hence, zero energy corresponds to equilibrium point, v=0 and $\dot{v}$=0. Instability

corresponds to a growth in mechanical energy while asymptotic stability ensures the convergence of mechanical energy to zero. Differentiating V with respect to time (assuming $M = M^T > 0$) yields:

$$\dot{V} = \dot{\eta}^T [M(\eta)\ddot{\eta} + g(\eta)] + \frac{1}{2}\dot{\eta}^T M(\eta)\dot{\eta} \tag{3.40}$$

The expression for $\dot{V}$ can be rewritten as

$$\dot{V} = \dot{\eta}^T [M(\eta)\ddot{\eta} + C(v,\eta)\dot{\eta} + g(\eta)] + \frac{1}{2}\dot{\eta}^T [M - 2C(v,\eta)]\dot{\eta} \tag{3.41}$$

By applying the skew-symmetric property: $\dot{\eta}^T (M - 2C)\dot{\eta} = 0 \forall \dot{\eta}$,

$$\dot{V} = \dot{\eta}^T [M(\eta)\ddot{\eta} + C(v,\eta)\dot{\eta} + g(\eta)] \tag{3.42}$$

In controls-fixed stability analysis, the dynamics of the control inputs has been neglected. Hence we can consider the system:

$$M(\eta)\ddot{\eta} + C(v,\eta)\dot{\eta} + D(v,\eta) + g(\eta) = 0 \tag{3.43}$$

Applying this equation to the expression for $\dot{V}$,

$$\dot{V} = -\dot{\eta}^T D\dot{\eta} = v^T Dv \tag{3.44}$$

According to Lyapunov's stability theory, sufficient conditions for control-fixed stability are:

(i) V>0 for all $\dot{\eta}, \eta \in R^n$, whereas $\dot{\eta} \neq 0$ and $\eta \neq 0$. Hence, $\dot{\eta}^T M(\eta)\dot{\eta} = v^T Mv > 0,\ v \neq 0$ if and only if the inertia matrix : M>0

(ii) $\dot{V} < 0$ for all $v \in R^n$ if and only if the damping matrix: $D(v) > 0 \forall v \in R^n$

(iii) $V \to \infty$ as $\|\eta\| \to \infty$ and $\|\dot{\eta}\| \to \infty$

Among above three conditions, first one implies that the inertia including hydrodynamic added mass must be strictly positive. For underwater vehicles we can assume constant added mass (independent of the wave frequency) which implies that

$\dot{M} = 0$ and $M = M^T > 0$. The second condition simply states that the system must be dissipative which is also true for uncontrolled undisturbed underwater vehicle.

The uncontrolled nonlinear system can be considered as autonomous since it does not explicitly depend on time t and can be expressed as $\dot{x} = f(x)$ (3.45)

Therefore, Lyapunov's direct method can be successfully used to prove the stability of considered model of underwater robot. Stability of equilibrium points in the sense of Lyapunov: An equilibrium point is stable if all solutions starting at nearby points stay nearby; otherwise it is unstable. It is asymptotically stable if in addition, all solutions tend to the equilibrium point as time approaches infinity.

3.5 Analysis on Three-dimensional path planning

In present research work, a small size underwater robot (Figure 3.1) has been employed for motions in surge, heave and yaw directions within calm water. For localizing robot in 3-D workspace, some conventional process for measuring distance and orientation are required to be followed, such as, Euler distance between any two consecutive positions of robot ($robx_i$, $roby_i$, $robz_i$) and ($robx_{i-1}$, $roby_{i-1}$, $robz_{i-1}$) in 3D environment can be formulated as, $dist_i = \sqrt{(\text{rob}x_i - \text{rob}x_{i-1})^2 + (\text{rob}y_i - \text{rob}y_{i-1})^2 + (\text{rob}z_i - \text{rob}z_{i-1})^2}$ (3.46)

and Euler angle between the axis of robot's body frame and assumed straight line from the robot's current pose ($robx_i$, $roby_i$, $robz_i$) to the next possible pose ($robx_{i-1}$, $roby_{i-1}$, $robz_{i-1}$) in 3D workspace (as shown in Figure 3.4) can be computed as,

$\psi_i = \tan^{-1}(\frac{roby_i - roby_{i-1}}{robx_i - robx_{i-1}})$ in horizontal plane

and $\theta_i = \tan^{-1}(\frac{robz_i - robz_{i-1}}{\sqrt{(robx_i - robx_{i-1})^2 + (roby_i - roby_{i-1})^2}})$ in vertical plane (3.47)

Traditionally, rotation of an underwater system in horizontal plane has been kept bounded within $-\pi \leq \psi < \pi$. To maintain a stable horizontal posture within water, rotation in vertical plane or roll angle will be constrained to the range of $-\pi/2 < \theta < \pi/2$.

To trace a desirable path, navigational controller embedded with underwater robot must decide suitable heading angle for underwater motion in horizontal or vertical plane with respect to the onboard sensors' data about distances and orientations of robot's current

pose from the specified target and nearest obstacles. Thrusters of underwater robot will be actuated as per decisions taken by navigational controller. On detection of obstacles, underwater robot must change its direction of motion or heading angle to avoid collisions with obstacles as shown in Figure 3.4. For no obstacle case, underwater robot will be directed towards the target. Heading angle of the robot must be updated in an online manner as robot moves from one position to another until the stopping criterion or goal has reached. Depending on the beam angle of on board sonar sensor, the measured heading angle will be within range of $(-60^0, 60^0)$ for θ_i and $(-90^0, 90^0)$ for ψ_i. In vertical plane, elevation angle (θ) is approximated as positive for above x-y plane and negative for below the x-y plane. In horizontal plane, negative sign of heading angle denotes that the robot has to take turn in clockwise and for positive sign robot will rotate in counter-clockwise direction. For no obstacle case, heading angle will be zero which means orientation of robot will be same as goal.

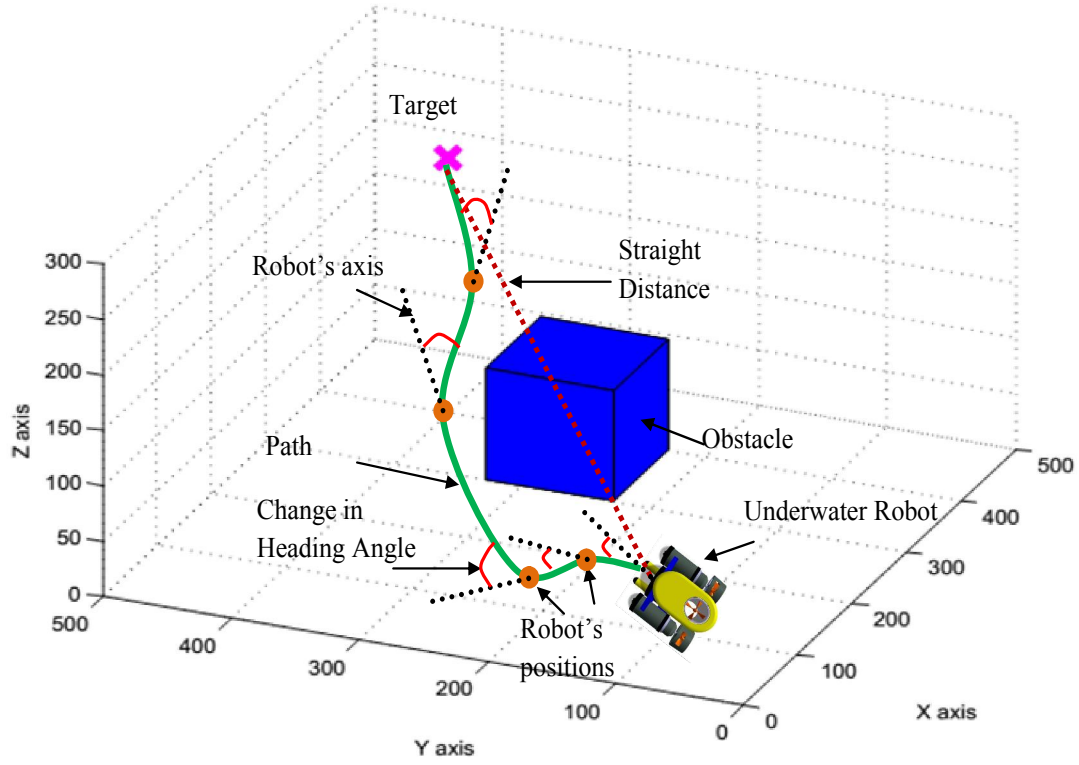


Figure 3.4: Variation in heading angle of underwater robot during navigation

Assumptions about underwater robot for current investigation can be listed as follows:

- The proposed underwater robot is able to do linear motion in forward or backward, up or down as well as turning in left or right in 3-D workspace of calm water.
- Complex dynamics of underwater robot are subjected to non-holonomic constraints which limits directions of motion in underwater environment.
- Comparatively small spinning radius of underwater robot within huge workspace may reduce the influence of non-holonomic constraints which may be created due to complex dynamics of three-dimensional motion.
- The speed of the robot relative to the underwater current is assumed to be constant.
- As dimension of underwater robot is also very small in comparison with much bigger workspace of real-time environment, so the assumption of the robot as a point without size for simulated environment is well justified.

3.6 Summary

In current research work, underwater robot is modelled as a mass point which is free to move in the 3D space, so the dynamics of the vehicle is not prioritized here. The major objectives are to develop a robust obstacle avoidance strategy and an optimal path from source to destination.

4. Analysis of Reactive Behaviors for Underwater Robot based on Manifold ANFIS Approach

Human perception based decision making ability has been employed in present chapter to find out heading angle of underwater robot. Easy understanding and interpretability of ANFIS has attracted present research work to implement it as path planning algorithm of underwater robot. Advantages of Fuzzy Inference System and Neural Network over complex computation of traditional algorithms have been incorporated in three-dimensional process.

4.1 Introduction

Both neural network and fuzzy logic are model-free estimators and share the common ability to deal with the uncertainties and noise [151]. Both of them encode the information in parallel and distribute architectures in a numerical framework. Hence, it is possible to convert fuzzy logic architecture to a neural network and vice versa. This makes it possible to combine the advantages of neural network and fuzzy logic. A network obtained in this way could use powerful training algorithms that neural networks have at their disposal, to obtain the required parameters not available in the fuzzy logic architecture. The ANFIS combines the two approaches, neural networks and fuzzy systems [77]. Combining these two intelligent approaches, good reasoning is achieved in quality and quantity. Researchers have observed [78-86] that the complexity increases with the increase in the number of inputs to ANFIS and the number of rules will be also increased exponentially (Page no.56). Present research work has focused on utilisation of ANFIS model for path planning of underwater robot in a cluttered environment. Fuzzy rules have been designed based on sensory inputs to deal with vagueness of environment. Simulation and experimental results have been executed for analysing performance of proposed ANFIS based navigational approach.

4.2 Manifold ANFIS Approach for Fusion of Reactive behaviors

During autonomous navigation, key reactive behaviours of underwater robot can be categorised as obstacle avoidance and target seeking in a broad manner. Considering both behaviours, manifold ANFIS approach has been implemented for approximating required amount of changes in rotation angles (for both horizontal and vertical planes (Figure 4.1). Each behavioural module is a combination of two ANFIS models which estimate the

required change in rotational angles of horizontal and vertical planes respectively. Online sensory information about robot's position and orientation with respect to nearest obstacle and target of given scenario have been considered as inputs for ANFIS models of these behavioural modules.

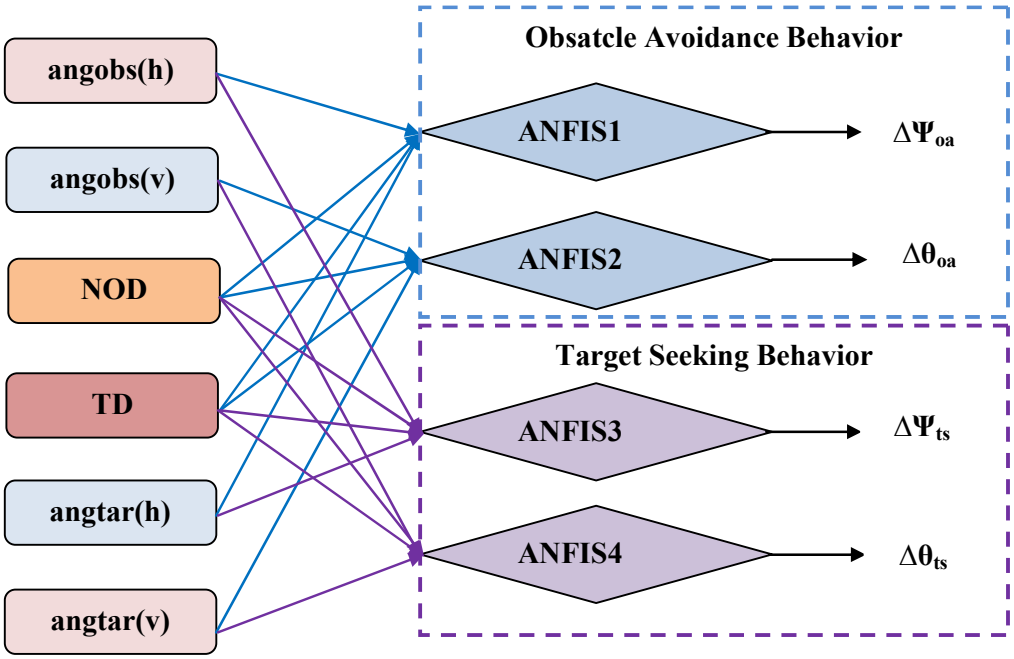


Figure 4.1: Block Diagram of Reactive Behaviors' Modules in terms of ANFIS Models

Fuzzy membership functions split the input dimension into many local regions based on either human perception or statistical distributions as per specification of particular problem [152]. Each fuzzy region can be parameterized through linear or nonlinear models such as triangular, trapezoidal, Gaussian etc. Present research work has favored bell shape membership function over other shapes as it has more controlling parameters than other linear functions. Bell shaped function (Figure 4.2), which can fuzzify the crisp values of inputs to the values between 0 and 1, can be represented by a nonlinear function as follows [153];

$$\mu_m(x_i) = \frac{1}{1+\left(\frac{x - c_{im}}{a_{im}}\right)^{2b_{im}}} \tag{4.1}$$

Where, x_i denotes crisp value of i^{th} input; $\mu_m(x_i)$ represents the fuizzified value of i^{th} input for its m^{th} region; a_{im}, b_{im} and c_{im} are the parameters for the fuzzy membership function.

Here, c_{im} signifies the centre of curve; a_{im} is the half width; and $b_{im}/2a_{im}$ restraints the slope at the crossover points [154]. By adjusting these three parameters, the shape of bell-shaped function can be altered to induce adaptive nature in fuzzy inference system.

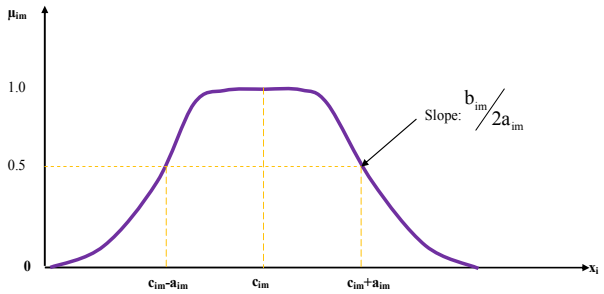


Figure 4.2: Bell shaped membership function

Three bell-shape membership functions overlapping with one another has been chosen here for each input variable to increase the approximation ability of ANFIS by actuating at least two local regions simultaneously for any input value. For proposed behavioural architecture, distribution of six inputs into their respective membership functions (labelled by linguistic terms) has been listed in Table 4.1. Pictorial view of membership function distribution for each input variable can be seen in Figure 4.

Table 4.1: Membership Functions for Input Variables

Input Variables	**Linguistic terms for Membership functions**		
Nearest obstacle Distance **(NOD)** and Target Distance**(TD)**:	Near	Medium	Far
Heading angles between robot and its nearest obstacles **(angobs(h)** and **angobs(v))** in the horizontal and vertical planes respectively and Target angles both in horizontal **(angtar(h))** and **vertical (angtar(v))** planes:	Rotate Clockwise (RCW)	No Rotation (NR)	Rotate Counter Clockwise (RCCW)

The objective of each behavioural module is to estimate the required change in rotation angles ($\Delta\Psi$ and $\Delta\theta$) for x-y and x-z plane respectively as shown in Figure 4.1. The outputs of two modules have been mingled together by utilizing weight factors. Some human perception based heuristic knowledge has also been incorporated in behavioural module design as follows:

(a) Collision avoidance has the higher priority and it can override other behaviours.

(b) If there is no obstacle between robot's current pose and target of the present scenario, its main reactive behaviour will be to steer towards target as soon as possible.

(c) Robot will choose path directed towards target over obstacles when obstacles and target are found to be in opposite sides of robot.

(d) For obstacle and the goal both in the same side of the robot, heading angle towards target will be modified as per ANFIS rules.

(e) When heading angle of robot towards nearest obstacle and target angle is almost same by value and sign both, robot has to take a turn as per rules for avoiding obstacle. After that, robot has to continue the previous heading direction towards target.

(f) If heading angle of robot towards nearest obstacle is very large but target angle is near about zero, then robot will choose to move directly toward the target.

A. Obstacle Avoidance Behaviour:

When the readings from sensors embedded in robot for obstacle detection are less than the minimum threshold values, the obstacle avoidance behavior has been required to be activated. Based on six inputs (as defined in Table 4.1), two ANFIS models of Obstacle avoidance behaviour module can produce required change in rotation angles for avoiding obstacles ($\Delta\Psi_{oa}$ and $\Delta\theta_{oa}$) in x-y and x-z plane respectively.

B. Target Seeking Behaviour:

If robot does not find any obstacles between robot's current pose and target of the present scenario, its main reactive behavior will be to steer towards target as soon as possible. The inputs for two ANFIS models in target seeking behaviour module (in Figure 4.1) are same as previous module. The outputs of this behavioural module can be denoted as required change in heading angles ($\Delta\Psi_{ts}$ and $\Delta\theta_{ts}$) for seeking target in x-y and x-z plane respectively.

C. Fusion of Multiple Behaviours:

The coordination of behaviours towards a single outcome has been achieved based on value of assigned weight factors (K_{oa} and K_{ts}) as a measure of impact for the current situation. The computation of total changes in rotation angle for underwater robot's current position in terms of $\Delta\Psi$ and $\Delta\theta$ has given below:

$$\Delta\psi = \frac{\frac{\Delta\psi_{oa}}{K_{oa}} + \frac{\Delta\psi_{ts}}{K_{ts}}}{K_{deno}} \; and \; \Delta\theta = \frac{\frac{\Delta\theta_{oa}}{K_{oa}} + \frac{\Delta\theta_{ts}}{K_{ts}}}{K_{deno}} \tag{4.2}$$

Where, $K_{deno} = \frac{K_{oa} + K_{ts}}{K_{oa} \bullet K_{ts}}$; K_{oa} and K_{ts} : Weight factors for two behavioural modules respectively.

Values for K_{oa} and K_{ts} can be chosen based on following hypotheses:

(a) Each obstacle is assumed to be surrounded by a sphere with certain diameter whose value is little more than highest dimension of that particular obstacle. For analysis purpose, each sphere is assumed to have 100 points on its surface which sensor of underwater robot can sense as obstacle points. Value of K_{oa} has been taken as inverse of number of obstacle points near about robot for its current position. So, Increase in intensity of obstacles in environment expects low value of K_{oa}.

(b) In other way, as number of targets or its positions are fixed in environment, value for K_{ts} will be constant irrespective of number of obstacles. For $K_{ts} > K_{oa}$, target seeking behaviour will come in first priority to reduce path length.

So, values of K_{oa} and K_{ts} have been dynamically adjusted throughout the navigation process. To find out suitable values of K_{oa} and K_{ts}, a trial and error analysis has been executed in simulation study on three-dimensional navigation. Such integration of robotic behaviours within rule base of ANFIS models may expand flexibility and robustness of ANFIS based navigational algorithm for underwater motion.

4.3 Framework of ANFIS Model

Reasoning power of fuzzy logic and learning ability of neural network have been integrated in ANFIS models which have been employed here to find out the precise change in heading angle of underwater robot for avoiding obstacles during motion through an impulsive underwater environment. Rules of an ANFIS model for four inputs and one output can be outlined in a generalized manner as follows [77]:

IF x1 is P_a, x2 is Q_b , x3 is R_c, x4 is S_d THEN $f_j = p_j$ x1 + q_j x2 + r_j x3 + s_j x4+ u_j

Where j = No. of rules; a, b, c, d = 1 to 3 as each input is fuzzified through three bell-shaped membership functions. P, Q, R and S are the fuzzy membership sets specified for

the input variables x1, x2, x3 and x4. For jth rule, f_j is linear combination of crisp value of inputs which are scaled by consequent parameters such as $\{p_j, q_j, r_j, s_j$ and $u_j\}$.

By adjusting these premise and consequent parameters, the desired output of the ANFIS controller can be obtained. ANFIS model has been represented each segment of first order Takagi-Sugeno Fuzzy system (e.g: fuzzification, rule base, inference mechanism and defuzzification) in terms of neural network layer. All nodes of a layer in the ANFIS model must have same transfer functions [154]. The output obtained from node function will be the input to the subsequent layer as shown in Figure 4.3. The complete flowchart of proposed ANFIS based underwater navigation has been depicted in Appendix-B.

Layer of Input Nodes: The input layer collects crisp data from arrays of sensors inputs regarding the distances and heading directions which may symbolize as $\{x_1, x_2, x_3$ and $x_4\}$.

Fuzzification Layer: Each node for this layer contains a particular bell shape membership function (or activation function) whose parameters can be altered as described in eq. 4.1. Node function stipulates the degrees to which the inputs gratify the quantifier. The output from each node of layer 2 can be specified as,

$$O_{2,m} = \mu_m(x_i) \tag{4.3}$$

Where, m=12: Total membership functions for four inputs or the number of nodes in layer 2 and x_i: i^{th} input to the m^{th} node of layer 2.

Layer for Rule-base: Due to absence of activation function, nodes of this layer can be considered as circular or fixed one (symbolized by "π")**.** Premise part of rules for Takagi- Sugeno fuzzy model have been represented by the nodes of Layer-3. The output of each node is the product of all incoming values i.e. firing strength (degree of fulfillment) of the associated rule. The output for j^{th} node of this layer is as follows:

$$O_{3,j} = w_j = \mu_{P_a}(x_1) * \mu_{Q_b}(x_2) * \mu_{R_c}(x_3) * \mu_{S_d}(x_4) \tag{4.4}$$

Where, j=81: Rule number, a, b, c, d = 1 to 3: Number of membership functions and P, Q, R and S are the fuzzy membership sets specified for the input variables x1, x2, x3 and x4.

Layer for Normalization: Each fixed node of this layer (labeled as "N") normalizes the firing strength of a given rule. Number of nodes in current layer is same as previous one. The output for j^{th} node can be expressed as,

$$O_{4,j} = \bar{w}_j = \frac{w_j}{\sum_{j=1}^{81} w_j} \tag{4.5}$$

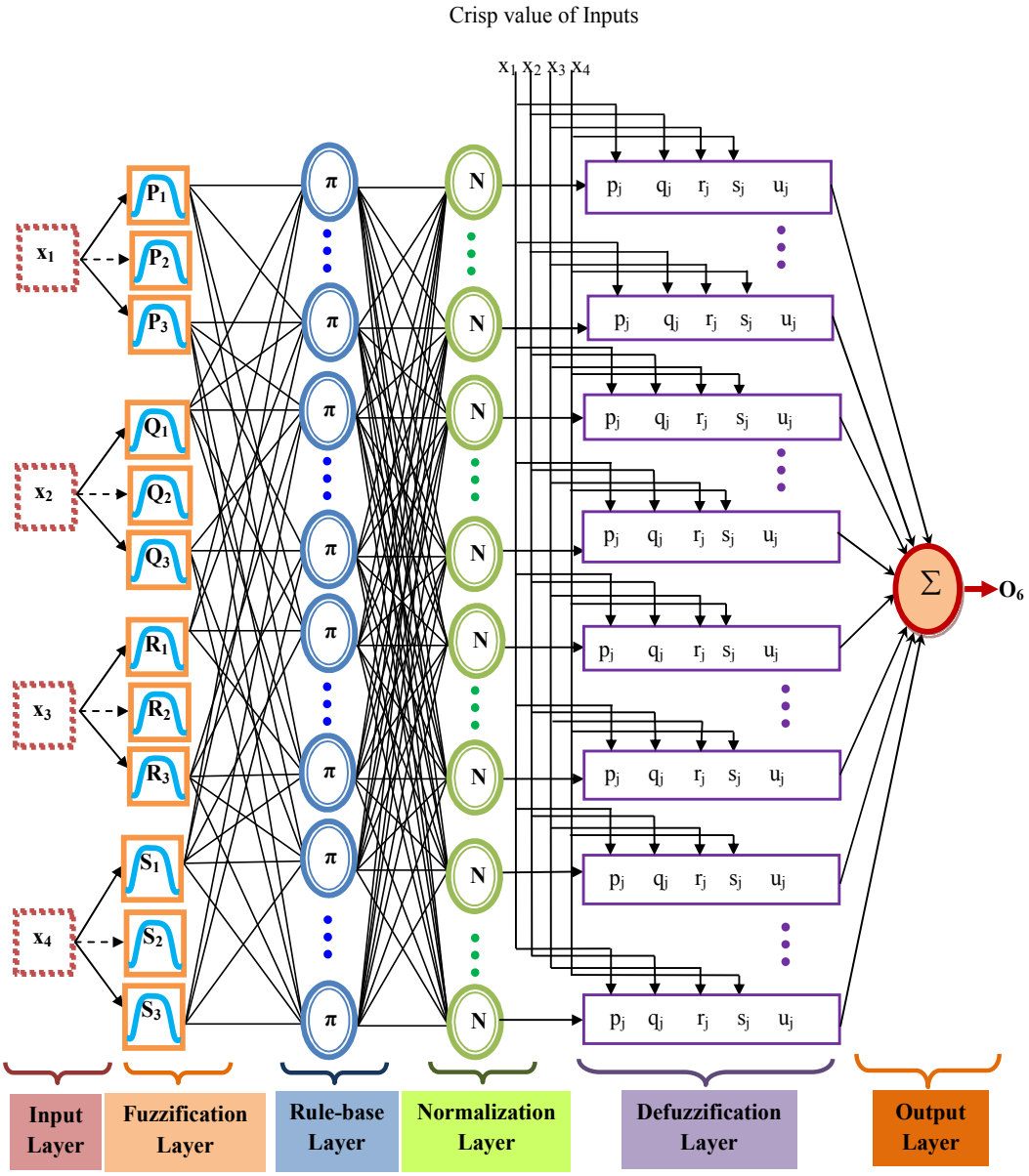


Figure 4.3: Six-layers ANFIS architecture for determining change in heading direction

Layer of Consequent Function: Here number of nodes is same as previous layer. Each node in this layer is adaptive by nature as output is the product of normalized firing strength and consequent portion of Takagi-Sugeno fuzzy rule as node function. For initial values of x_1, x_2, x_3 and x_4, the defuzzified output can be generalized as follows:

$$O_{5,j} = \bar{w}_j * f_j = \bar{w}_j * (p_j.x_1 + q_j.x_2 + r_j.x_3 + s_j.x_4 + u_j) \quad (4.6)$$

Where, $\bar{w}_j$ is a normalized firing strength or output of j^{th} node from previous layer and $\{p_j, q_j, r_j, s_j, u_j\}$ can be referred as consequent parameters of j^{th} rule.

Layer of Output Node: The single node of this layer is a fixed node (circular) labeled as "Σ", which summaries all defuzzified outcomes of previous layer into final crisp value of change in rotation angle (in degrees).

$$O_6 = \sum_{j=1}^{81} \bar{w}_j f_j = \frac{\sum_{j=1}^{81} w_j f_j}{\sum_{j=1}^{81} w_j} \quad (4.7)$$

In ANFIS, neural network learns fuzzy inference system's behavior from a large set of input-output training pairs by unsupervised learning and apply this knowledge to adaptively tune the parameters of FIS [28]. So, both training algorithm and dataset must be precisely designed to significantly reduce the difference between estimated and predicted output of ANFIS model.

4.4 Back Propagation based Learning Mechanism of ANFIS model

In assessment of parameters (both premise and consequence) of the model ANFIS, the hybrid-learning procedure [77] (back-propagation and least squares estimate) may take a long time to converge along with high complexity. To avoid this difficulty, only back-propagation algorithm for learning of parameters in both sides has been considered as an alternative measure. In a generalized manner, the training data set is assumed to have P entries. The error measure (or energy function) for the p^{th} ($1 \le p \le P$) learning pattern as per back-propagation rule [155]: $E_p = \frac{1}{2}(T_p - O_{l,p})^2$ (4.8)

Where, T_p is the targeted output and $O_{l,p}$ is the actual output of l^{th} layer $\{1 \le l \le 6\}$ of ANFIS model for p^{th} learning pattern.

Now, the partial derivatives of error function with respect to parameters (antecedent and consequent) of fuzzy systems which needs to be tuned have to be computed. For j^{th} rule parameter set $\rho_j \{\epsilon \rho_a, \rho_c\}$, $\frac{\partial E_p}{\partial \rho_j} = \frac{\partial E_p}{\partial O_{l,p}} \frac{\partial O_{l,p}}{\partial O_{l-1,p}} \frac{\partial O_{l-1,p}}{\partial \rho_j} = -(T_p - O_{l,p}) \frac{\partial O_{l,p}}{\partial O_{l-1,p}} \frac{\partial O_{l-1,p}}{\partial \rho_j}$ (4.9)

The overall error measure is: $E = \sum_{p=1}^{P} E_p$ (4.10)

By following the chain rule of back propagation algorithm, the partial derivatives of total error have been computed here in a sequential manner. As linear consequent functions are in 5^{th} layer, so, the error rate for consequent parameters (ρ_c) for p^{th} learning pattern can be as follows,

$$\frac{\partial E}{\partial \rho_c} = \frac{\partial E}{\partial O_6} \frac{\partial O_6}{\partial O_5} \frac{\partial O_5}{\partial \rho_c} \tag{4.11}$$

Applying equation (4.6) and (4.7), $\frac{\partial O_6}{\partial O_5} = \frac{\partial \left\{ \sum_{j=1}^{81} \overline{w}_j f_j \right\}}{\partial \left\{ \overline{w}_j * f_j \right\}} = 1$

(4.12)

Then, $\frac{\partial E}{\partial \rho_c} = \frac{\partial E}{\partial O_6} \frac{\partial O_5}{\partial \rho_c} = -(T - O_6) \frac{\partial O_5}{\partial \rho_c}$ (4.13)

Where, T is the desired outcome and O_6 is the output of final layer of ANFIS model for p^{th} learning pattern.

Considering set of consequent parameters $\rho_c \{p_j, q_j, r_j, s_m, u_m\}$ for j^{th} rule, $\frac{\partial O_5}{\partial \rho_c} = \frac{\partial O_5}{\partial p_j} + \frac{\partial O_5}{\partial q_j} + \frac{\partial O_5}{\partial r_j} + \frac{\partial O_5}{\partial s_j} + \frac{\partial O_5}{\partial u_m} = \overline{w}_j \bullet (x1 + x2 + x3 + x4 + 1)$ (4.14)

Combining equation (4.12), (4.13) and (4.14), the error rate for consequent parameters,

$$\frac{\partial E}{\partial \rho_c} = -(T - O_6) \bullet \overline{w}_j \bullet (x1 + x2 + x3 + x4 + 1) \tag{4.15}$$

Similarly, the error rate for set of antecedent parameters, $\rho_a \epsilon \{a_m, b_m, c_m\}$ for p^{th} learning pattern can be computed as,

$$\frac{\partial E}{\partial \rho_a} = \frac{\partial E}{\partial O_6}\frac{\partial O_6}{\partial O_5}\frac{\partial O_5}{\partial O_4}\frac{\partial O_4}{\partial O_3}\frac{\partial O_3}{\partial O_2}\frac{\partial O_2}{\partial \rho_a} \tag{4.16}$$

Applying equation (4.6) & (4.7) for j[th] node of layer 4 and 5,

$$\frac{\partial O_5}{\partial O_4} = \frac{\partial\left\{\overline{\mathrm{w}}_j\ \mathrm{f_j}\right\}}{\partial \overline{\mathrm{w}}_j} = \mathrm{f_j} \tag{4.17}$$

From equation (4.4) & (4.5) for j[th] node of layer 3 and 4,

$$\frac{\partial O_4}{\partial O_3} = \frac{\partial\left\{\overline{\mathrm{w}}_j\right\}}{\partial w_j} = \frac{\sum_{j=1}^{81} w_j - w_j}{\left(\sum_{j=1}^{81} w_j\right)^2} \tag{4.18}$$

From equation (4.3) & (4.4) for m[th] node of layer 2 and j[th] node of layer 3,

$$\begin{aligned}\frac{\partial O_3}{\partial O_2} &= \frac{\partial w_j}{\partial \mu_m(x)} = \frac{\partial\left\{\mu_{\mathrm{P}_a}(x) * \mu_{\mathrm{Q}_b}(x) * \mu_{\mathrm{R}_c}(x) * \mu_{\mathrm{S}_d}(x)\right\}}{\partial \mu_m(x)} \\ &= w_j\left(\frac{1}{\mu_{\mathrm{P}_a}(x)} + \frac{1}{\mu_{Q_b}(x)} + \frac{1}{\mu_{R_c}(x)} + \frac{1}{\mu_{S_d}(x)}\right)\end{aligned} \tag{4.19}$$

Partial derivative of O_2 (output from layer 2) with respect to antecedent parameters $\rho_a \in \{a_n, b_n, c_n\}$ can be derived separately for three parameters,

$$\frac{\partial O_2}{\partial \rho_a} = \frac{\partial O_2}{\partial a_n} + \frac{\partial O_2}{\partial b_n} + \frac{\partial O_2}{\partial c_n} \tag{4.20}$$

$$\frac{\partial O_2}{\partial a_n} = \frac{\partial\left\{\dfrac{1}{1+\left\{\left(\dfrac{x-c_n}{a_n}\right)^2\right\}^{bn}}\right\}}{\partial a_n} = \begin{cases} \dfrac{2\left(\dfrac{x-c_n}{a_n}\right)^{2b_n} \cdot b_n \cdot (x-c_n)}{\left[1+\left(\dfrac{x-c_n}{a_n}\right)^{2b_n}\right]^2 \cdot {a_n}^2 \cdot \dfrac{x-c_n}{a_n}}, if\ \dfrac{x-c_n}{a_n} > 0 \\ \dfrac{-2\left(\dfrac{x-c_n}{a_n}\right)^{2b_n} \cdot b_n \cdot (x-c_n)}{\left[1+\left(\dfrac{x-c_n}{a_n}\right)^{2b_n}\right]^2 \cdot {a_n}^2 \cdot \dfrac{x-c_n}{a_n}}, if\ \dfrac{x-c_n}{a_n} < 0 \\ 0, if\ \dfrac{x-c_n}{a_n} = 0 \end{cases} \tag{4.21}$$

In a similar way, $\frac{\partial O_2}{\partial b_n}$ and $\frac{\partial O_2}{\partial c_n}$ can also be find out.

So, eq. 4.16 can be found out by combining equations 4.12, 4.13, 4.17, 4.18, 4.19, 4.20 and 4.21. After computing the error rate for both antecedent and consequent parameter set, the updated formula of the parameter set ρ_m $\{\rho_a, \rho_c\}$ can be denoted as per convention:

$$\Delta\rho_j = -\lambda \frac{\partial E}{\partial \rho_j} \tag{4.22}$$

Where λ is a learning or adaptation rate which can be conveyed as:

$$\lambda = \frac{s}{\sqrt{\sum_{j=1}^{81}\left(\frac{\partial E}{\partial \rho_j}\right)^2}} \tag{4.23}$$

Where, s is step size, the length of each gradient transition in the parameter space. The value of s can be changed to vary speed of convergence.

Performance of training algorithm solely depends on the desirable versatility presents within training and testing data sets. A data base of 500 input-output pairs has been prepared based on expert's knowledge or real-time experiment of underwater navigation to train four ANFIS models of proposed behavioural architecture. 70% of available data set (350) has been randomly chosen as training patterns to explore the search space properly and remaining 30% of data set (150) has been employed as testing patterns to verify the efficiency of developed ANFIS architecture. From database, only 50 patterns have been randomly chosen to be viewed as examples. 35 patterns have been employed for training purpose (as listed in Table 4.2) and rest has been used in testing purpose (Table 4.3). Different values of input variables for training patterns have also been visualized in Figure 4.4. The testing data set must be different from data set used in the training period to maintain generalization capability of proposed architecture. Based on the uploaded XL sheets of training and test data sets, the T-S model based FIS will be generated in MATLAB by following ANFIS architecture of Figure 4.2.

Table 4.2: Few examples of Training Patterns

Sl. No.	Inputs from Sensors						Output	
	NOD (cm)	**TD (cm)**	**angobs(h) (degree)**	**angobs(v) (degree)**	**angtar(h) (degree)**	**angtar(v) (degree)**	**$\Delta\Psi$ (degree)**	**$\Delta\theta$ (degree)**
1	20	130	-45	20	-60	45	-5	11.45
2	50	70	30	-30	-45	10	-4.78	4.29
3	90	160	60	-45	90	-30	12.39	26.35
4	40	180	-30	20	-20	0	-15.47	-17.23
5	60	150	45	10	0	30	3.51	-8.09
6	30	63	10	-20	-43	10	-14.3	19.47
7	70	120	30	15	75	25	7.5	-13.09
8	80	190	-20	30	10	80	10.89	21.14
9	50	85	-45	10	-30	30	-2.07	7.96
10	20	77	-60	-40	-90	-25	8.34	-15.75
11	40	110	15	15	45	-10	-21	9.83
12	65	90	120	40	140	0	-15.07	3.64
13	37	59	-30	15	-35	-30	9.41	24.05
14	43	87	-50	30	-15	45	14.35	-12.23
15	21	33	30	-20	45	-10	-10.23	-16.56
16	69	197	60	35	120	30	-2.59	7.89
17	13	56	-45	-60	-20	-45	5.87	-13.52
18	47	139	-90	-25	-60	-40	-13.09	27.29
19	69	170	45	30	120	10	7.21	-15.77
20	83	125	20	10	30	-20	3.56	2.63
21	34	69	150	15	110	40	4.25	10.95
22	47	95	-45	-20	20	-45	-5	11.45
23	26	81	-30	20	-45	5	-4.78	4.29
24	73	184	60	-15	90	-25	12.39	26.35
25	97	131	-90	30	-120	-10	-15.47	-17.23
26	51	98	-45	45	20	30	-2.07	7.96
27	30	173	35	-30	10	0	8.34	-15.75
28	65	90	120	40	140	0	-15.07	3.64
29	37	59	-30	15	-35	-30	9.41	24.05
30	43	87	-50	30	-15	45	14.35	-12.23
31	21	33	30	-20	45	-10	-10.23	-16.56
32	69	197	60	35	120	30	-2.59	7.89
33	13	56	-45	-60	-20	-45	5.87	-13.52
34	47	139	-90	-25	-60	-40	-13.09	27.29
35	53	86	81	45	123	0	-15.07	3.64

During training of ANFIS model in MATLAB environment, 200 iterations (epochs) have been found to be adequate for convergence of error towards a negligible value. Changes in parameters of membership functions of input variables through training can be viewed in Figure 4.5.

Table 4.3: Few examples of Testing Patterns

Sl. No.	Inputs from Sensors						Output in Horizontal Plane	Output in Vertical Plane
	NOD (cm)	TD (cm)	angobs(h) (degree)	angobs(v) (degree)	angtar(h) (degree)	angtar(v) (degree)	ΔΨ (degree)	Δθ (degree)
1	20	130	-45	20	-60	45	-6.9	16.26
2	50	70	30	-30	-45	10	-11.47	-9.71
3	90	160	60	-45	90	-30	15.23	-15.84
4	40	180	-30	20	-20	0	7.94	-4.59
5	60	150	45	10	0	30	-21.56	23.65
6	30	63	10	-20	-43	10	-13.42	-17.33
7	70	120	30	15	75	25	18.05	14.58
8	80	190	-20	30	10	80	5.79	6.39
9	50	85	-45	10	-30	30	-4.53	5.87
10	20	77	-60	-40	-90	-25	-8.67	-5.23
11	40	110	15	15	45	-10	-13.09	27.29
12	65	90	120	40	140	0	7.21	-15.77
13	37	59	-30	15	-35	-30	3.56	2.63
14	43	87	-50	30	-15	45	4.25	10.95
15	21	33	30	-20	45	-10	3.43	2.91

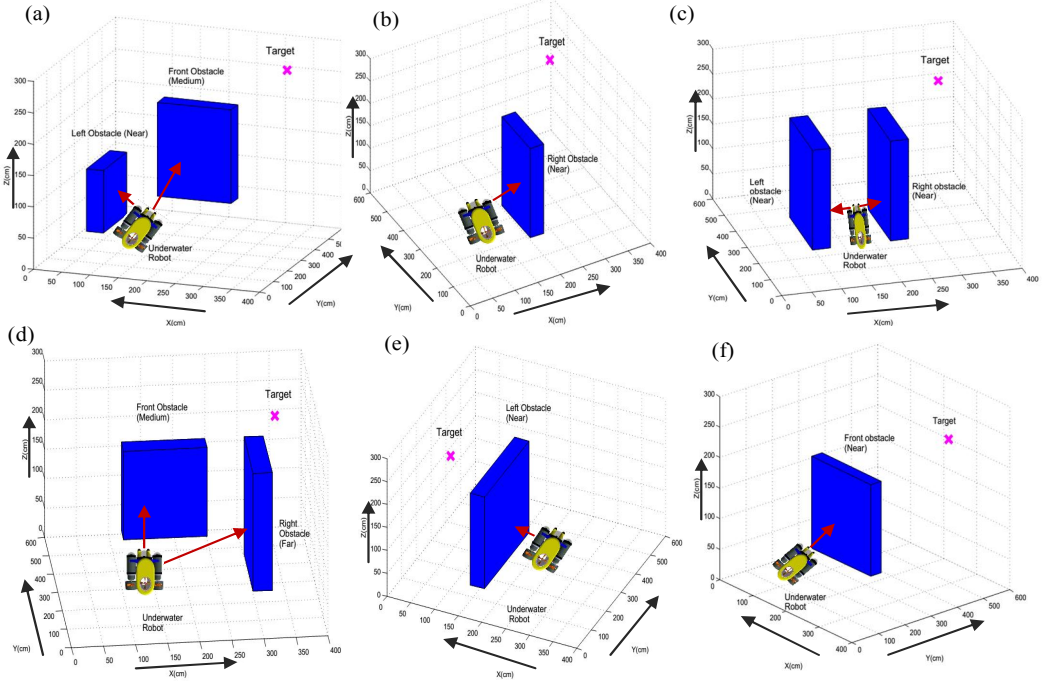


Figure 4.5: Visual examples of few training patterns

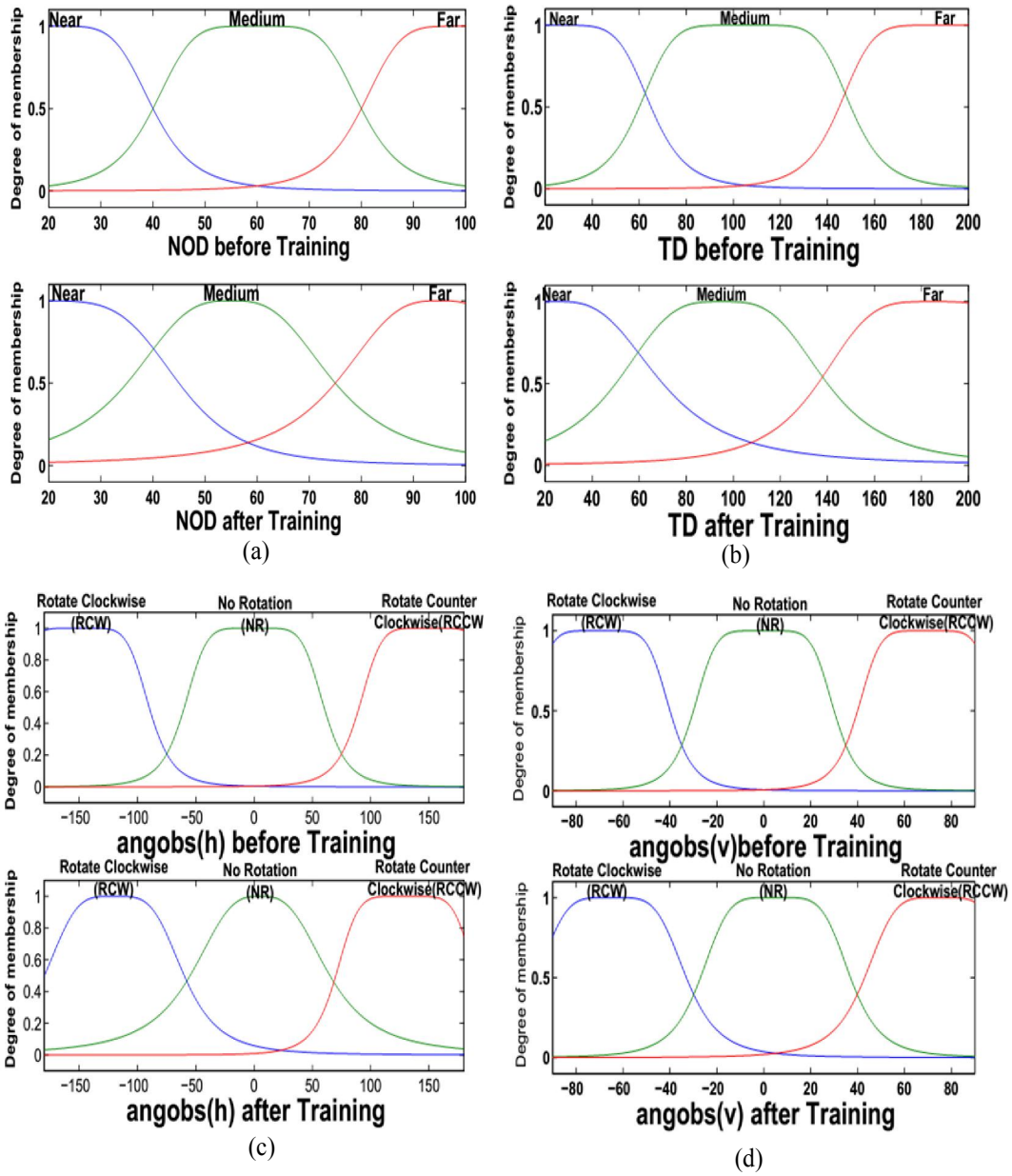


Figure 4.6: Membership Functions for Input Variables before and after training: (a) NOD (Nearest obstacle Distance), (b) TD (Target Distance), (c) angobs(h) (Obstacle angle in horizontal plane), (d) angobs(v) (Obstacle angle in vertical plane)

Note: Changes in parameters of membership functions for angtar(h) and angtar(v) will be same as angobs(h) and angobs(v) respectively.

4.5 Simulation Results

To evaluate the navigational performance of ANFIS based behavioral architecture, underwater environment has been virtually replicated as three-dimensional simulated

scenario of MATLAB. Before applying proposed ANFIS based navigational strategy in simulated environment, some assumptions on underwater robot and simulation environment have been considered here:

- As dimension of underwater robot is also very small in comparison with much bigger workspace of real-time environment, so the assumption of the robot as a point without size for simulated environment is well justified.
- The speed of the robot relative to the underwater current is assumed to be constant.
- Start and target coordinates have been assumed to be known for any simulated environment in MATLAB version R2008a.
- Same as real-time environment, from starting point onwards proposed path planning algorithm will continuously measure the distances and heading angles for robot's current position within simulated scenario until the goal has reached.
- When nearest obstacle distance is less than the threshold value, the robot has to steer at a certain angle (in horizontal or vertical plane) while keeping in mind target distance and angle as well.

By following the above assumptions, reactive behaviours of underwater robot has been verified while tracing a path from start to goal point within simulation environment which contains randomly distributed static obstacles. Collision avoidance has the utmost urgency and therefore, it can override other behaviors as shown in Figure 4.6.

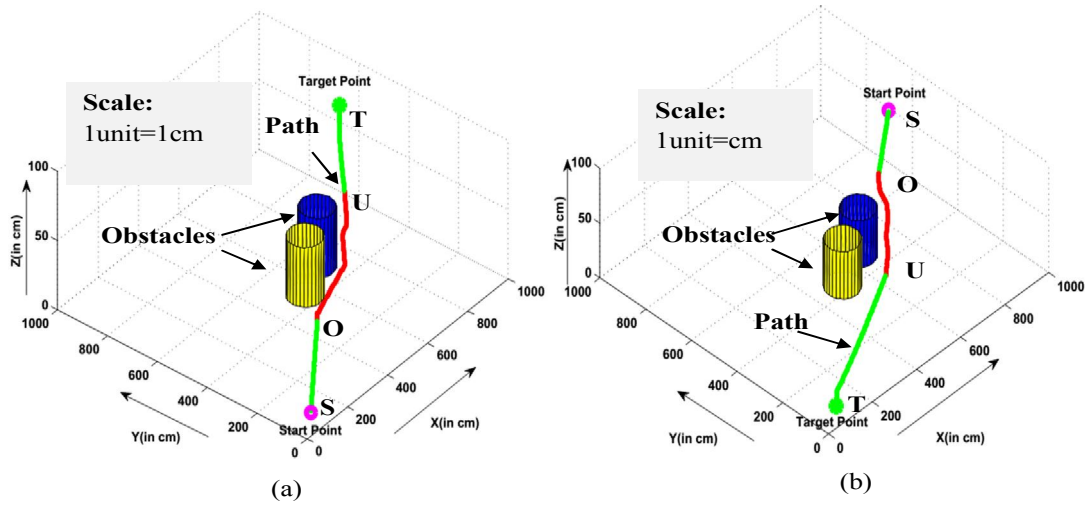


Figure 4.7: Simulation result for Reactive behaviours using Manifold ANFIS approach

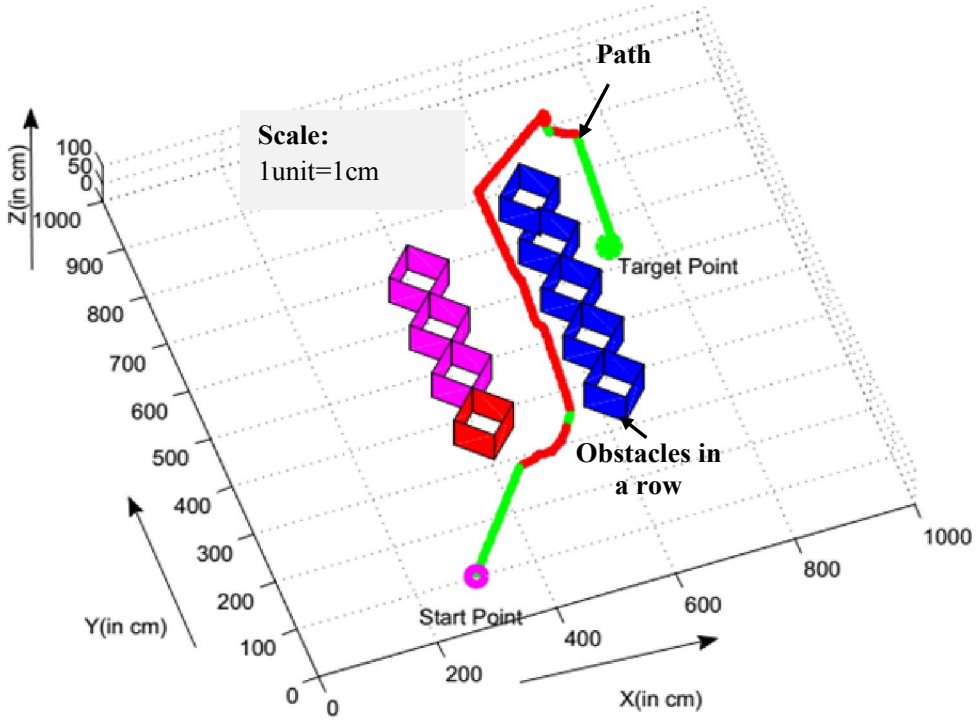


Figure 4.8: Wall following behaviour of underwater robot using Manifold ANFIS approach

Initial assumption is that underwater robot is unaware about the location of obstacles. Only starting point S (107, 70, 0) and target point T (853, 860, 90) are known positions. While moving towards target, robot has sensed the obstacle at point 'O' and avoided the nearest obstacle by changing robot's heading angle as estimated by ANFIS based behavioural architecture. The procedure for collision avoidance using navigational algorithm has been continued up to point U. After that as no obstacle is there, robot directly follows path towards target. In simulated path, changes in colour symbolize the activation of reactive behaviours by underwater robot. Altering the coordinates of source and target position within same simulation environment, performance has been analysed in Figure 4.6(b). Difference between navigational performances of Figures 4.6(a) and 4.6(b) regarding path length and execution time has been found to be negligible.

Underwater robot may trap in a local minima situation, in that case wall following behavior should be performed first, the robot must rotate clockwise or counterclockwise such that it can align and move along the wall. Such situations are already incorporated in training patterns of ANFIS models so that robot has been successfully performed this

particular behavior. Figure 4.7 shows the wall following Behaviour of underwater robot with start at (300, 100, 10) and the target position at (710, 630, 90) with the unknown environment. Increasing the number of obstacles and distributing them in different positions within simulated workspace, manifold ANFIS approach has been successfully implemented for navigation of underwater robot in Figure 4.8. Simulations have been performed for twenty times for each simulation environments same as Figures 4.6, 4.7, and 4.8. Among all of them the most viable paths traced by underwater robot with minimum path length have been viewed here.

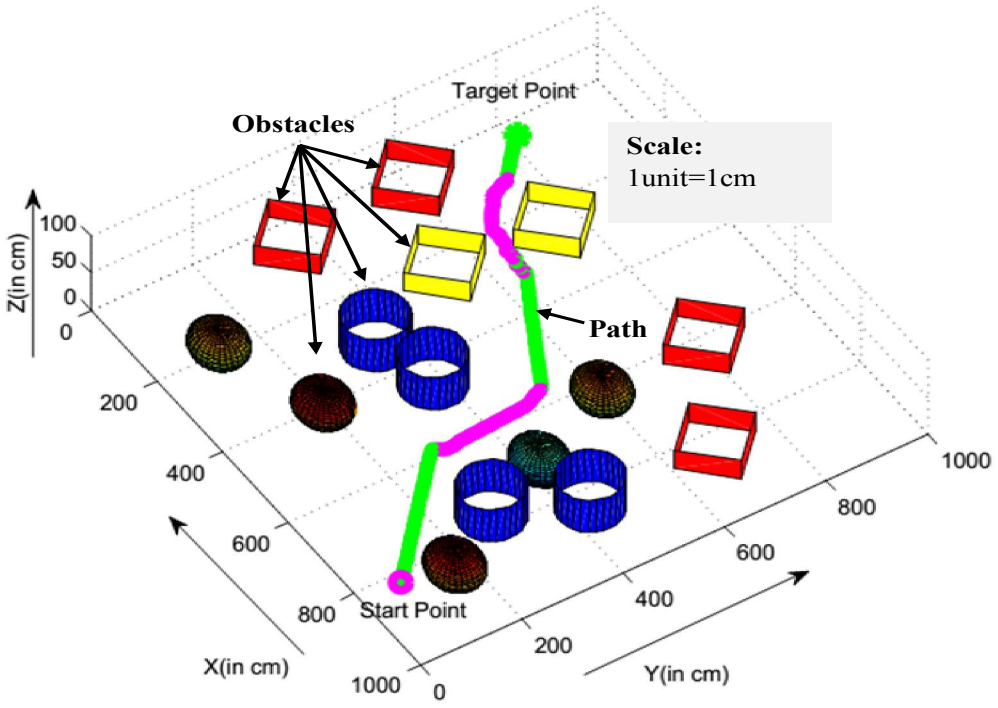


Figure 4.9: Simulated path for underwater robot in a cluttered environment using Manifold ANFIS approach

4.5.1 Impact of Weight Parameters

The weight factors K_{oa} and K_{ts} can be regulated to get better navigational performance within its work space. For different obstacle positions in between starting and target points, value of K_{oa} will be keep on changing as robot will move forward from start position towards target while avoiding obstacles. For each simulation K_{oa} starts with zero value which activates target seeking behaviour. As robot moves directly towards target, it may sense obstacle points within its threshold range, which results in low K_{oa}. Reduction in K_{oa} continues when robot moves through the area densely populated with obstacle

points as given in Table 4.4 for simulation result (Figure 4.9). In reverse way, when robot is near about target and obstacle points are also less in number K_{oa} will again slowly reduce to zero. K_{ts} is required to be constant throughout navigation as start and target positions are fixed. Change in K_{ts} directly influences the performance regarding path length and collision avoidance as well. Depending on values of K_{ts} four cases are listed in Table 4.4 for simulation study in Figure 4.9, such as, Case 1 (for $10 \le K_{ts} \le 90$), Case 2 (for $1 \le K_{ts} \le 9$), Case 3 (for $0.1 \le K_{ts} \le 0.9$) and Case 4 (for $0.01 \le K_{ts} \le 0.09$).

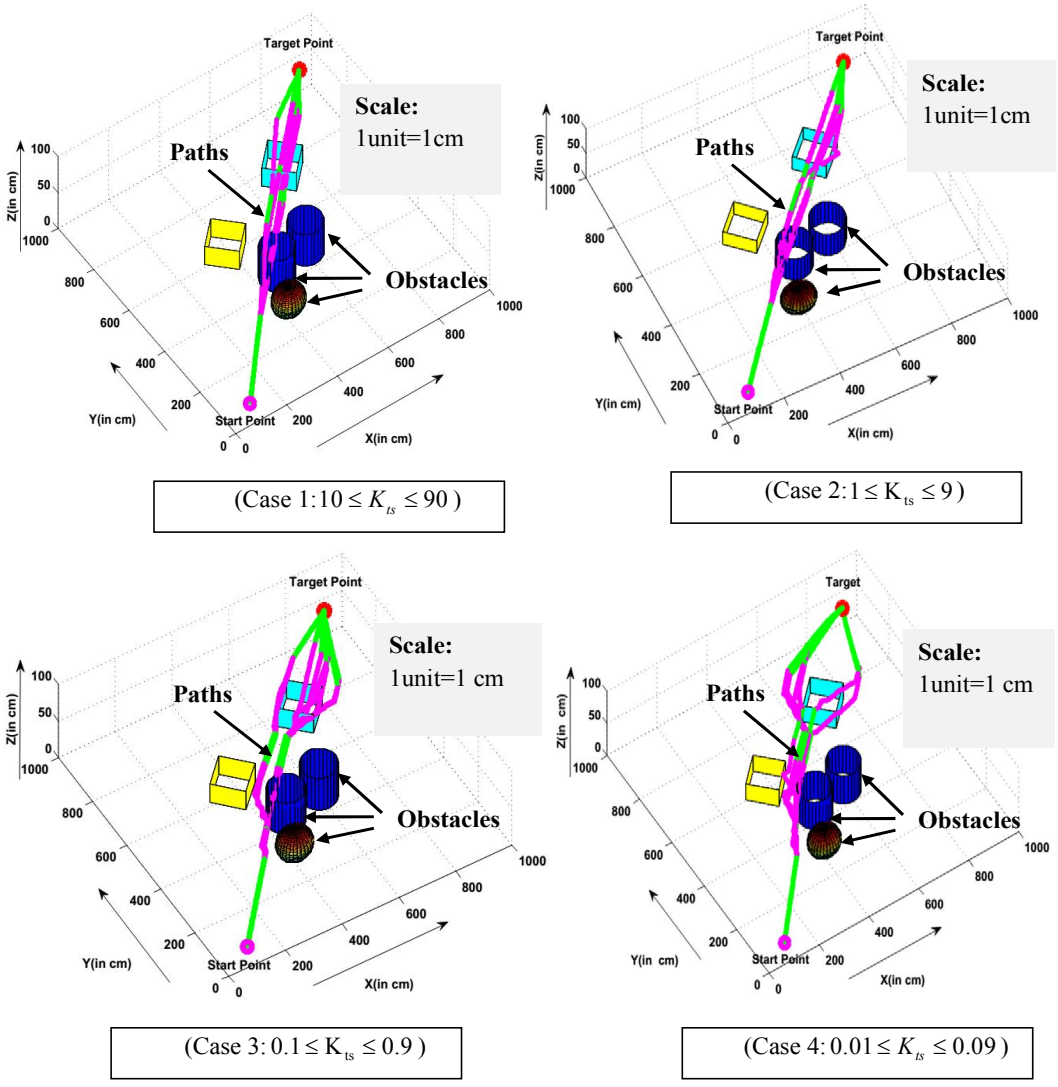


Figure 4.10: Path drawn by underwater robot for different values of K_{oa} and K_{ts} with same obstacles arrangement: Case (1) for $10 \le K_{ts} \le 90$; Case (2) for $1 \le K_{ts} \le 9$; Case (3) for $0.1 \le K_{ts} \le 0.9$ and Case (4) for $0.01 \le K_{ts} \le 0.09$

Table 4.4: K_{oa} and K_{ts} in different values for simulation environment as Figure 4.10

Case Number	K_{oa} (Different values during navigation from start to goal)	K_{ts} (Constant during navigation from start to goal)	Path length (in cm)	Collision with obstacles (Yes/No)
1	0.13, 0.26, 0.09, 0.057, 0.036, 0.019, 0.008, 0.003, 0.015, 0.027, 0.031, 0.022, 0.005, 0.072, 0.086, 0.097, 0.18, 0.1	10	1023	Yes (Indefinite motion)
		30	1011	Yes (Indefinite motion)
		50	963.6	Yes
		70	987	Yes
		90	951.2	Yes
2	0.17, 0.23, 0.05, 0.063, 0.039, 0.025, 0.004, 0.009, 0.018, 0.021, 0.043, 0.021, 0.012, 0.039, 0.051, 0.087, 0.13, 0.1	1	879	Yes
		3	835	Yes
		5	827.5	Yes
		7	841.3	Yes
		9	854	Yes
3	0.11, 0.37, 0.084, 0.043, 0.022, 0.014, 0.007, 0.002, 0.019, 0.033, 0.045, 0.018, 0.006, 0.035, 0.054, 0.083, 0.19, 0.1	0.1	785.9	Yes
		0.3	756	No
		0.5	773.2	Yes
		0.7	769.5	Yes
		0.9	757.3	No
4	0.15, 0.27, 0.07, 0.053, 0.032, 0.015, 0.009, 0.005, 0.011, 0.023, 0.035, 0.028, 0.007, 0.032, 0.056, 0.097, 0.15, 0.1	0.01	755.8	No
		0.03	761	Yes
		0.05	749.3	No
		0.07	743.6 (min.)	No
		0.09	748.9	No

By observing Table 4.4 and Figure 4.9, it has been found that few values of K_{ts} in case 4 can provide satisfactory navigational performance with collision free path. Values of K_{ts} in other three cases fail to avoid collision with obstacles and path lengths are also high. So, more simulations are required to perform for $0.01 \leq K_{ts} \leq 0.09$ to find out the approximate value of K_{ts} which can improve the navigational performance. In Figure 4.10, simulations have been performed by varying K_{ts} from 0.01 to 0.095 as given in Table 4.5. Values of K_{oa} have been varied throughout iterations. By analysing simulation result of Figure 4.10, Table 4.5 exhibits that the underwater robot follows the shortest path length for K_{ts} = 0.065. For K_{ts} =0.001 underwater robot may not reach the target as shown in Figure 4.11, so values of K_{ts} ($\in$ 0.001 to 0.009) have not been investigated.

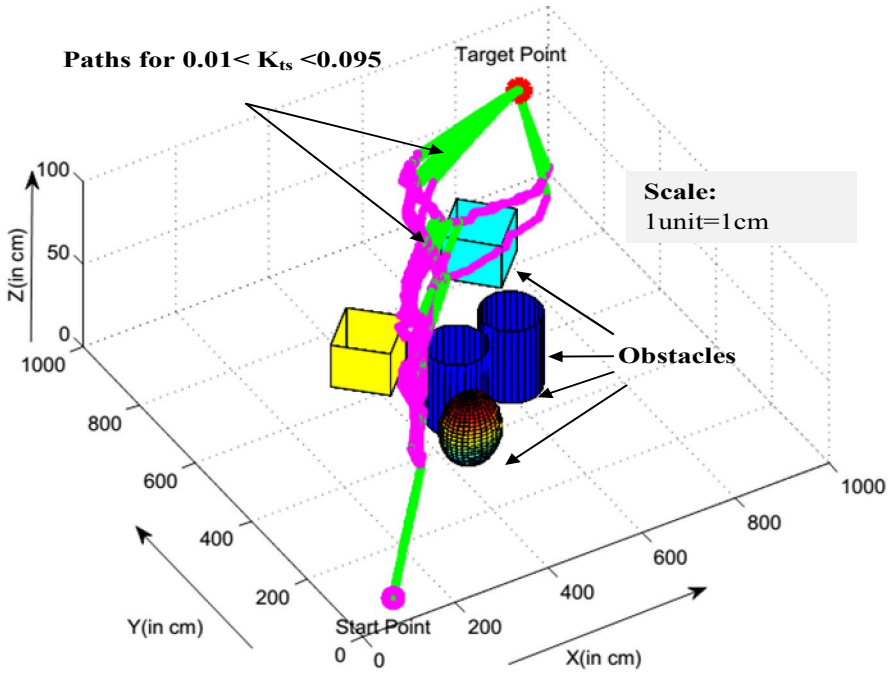


Figure 4.11: Paths followed by underwater robot for different values of K_{oa} and K_{ts} as analysed in Table 4.5

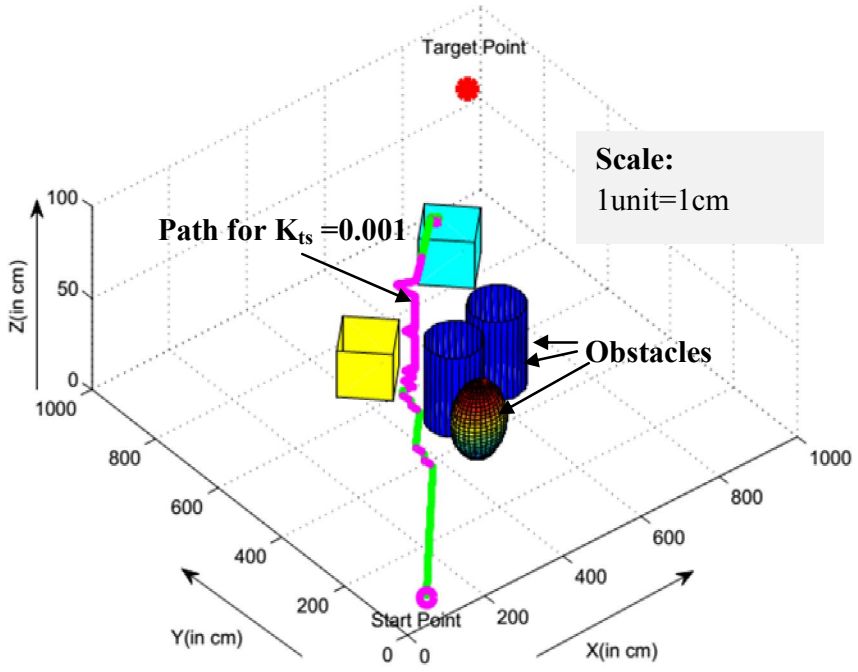


Figure 4.12: Path traced by underwater robot for specific K_{ts} value

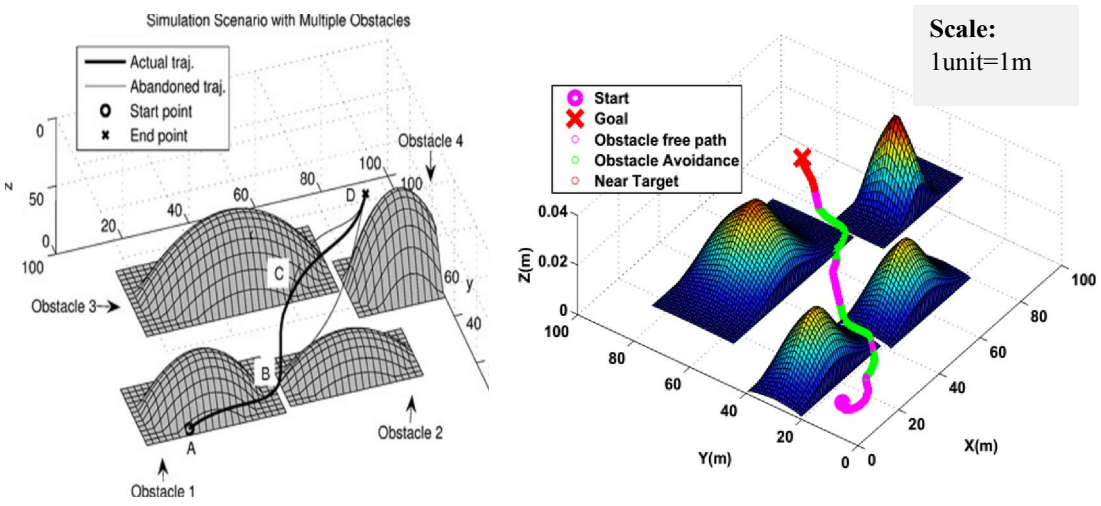
Table 4.5: Different values K_{oa} and K_{ts} for simulation environment such as Figure 4.11

K_{oa} (Different values during navigation from start to goal)	K_{ts} (Constant during navigation from start to goal)	Path length (in cm)	Collision with obstacles
0.15, 0.27, 0.07, 0.053, 0.032, 0.015, 0.009, 0.005, 0.011, 0.023, 0.035, 0.028, 0.007, 0.032, 0.056, 0.097, 0.15, 0.1	0.01	755.8	No
	0.015	757.3	No
	0.02	752.7	No
	0.025	760.3	Yes
	0.03	761	Yes
	0.035	758.2	No
	0.04	757.8	No
	0.045	751.6	No
	0.05	749.3	No
	0.055	748.7	No
	0.06	745.9	No
	0.065	741.3(min.)	No
	0.07	743.6	No
	0.075	744	No
	0.08	744.7	No
	0.085	746.3	No
	0.09	748.9	No
	0.095	747.5	No

4.5.2 Comparative study with other three-dimensional navigational approaches

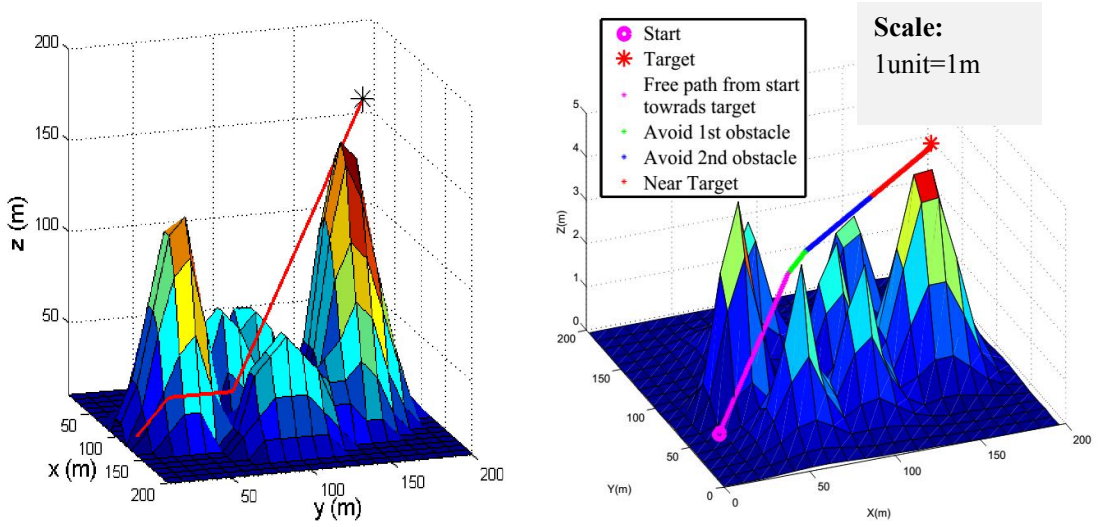
In this section previous researches on underwater navigation in simulated environment have been discussed and compared with the current approach. The comparative study can authenticate the proposed navigational strategy in a realistic manner. The details of comparison have been illustrated as follows:

(a) A motion planning approach to get collision free trajectory for AUVs in 3D unknown space has been designed by Yuan and Qu [156]. By reducing dimension of problem from 3D to 2D and also considering vehicle's kinematic model, collision has been prevented successfully in simulated environment. Keeping almost same obstacle arrangement as Figure 4.12 (a), manifold ANFIS approach has been applied in simulated environment. Sampling period is also one second same as Figure 4.12(a). Simulated path traced by ANFIS based behavioural architecture (Figure 4.12 (b)) has shown enough clearance from obstacles. Computational time used in Figure 4.12 (b) is only 12.73 sec in comparison with Figure 4.12 (a) where travel time is 25sec.



(a) (b)

Figure 4.13: (a) Simulated path in 3-D environment using collision-free approach by Yuan and Qu [156]; (b) Result provided by implementing ANFIS approach in a manifold manner



(a) (b)

Figure 4.14: (a) Path planning result in a 3-D complex environment without currents by Li and Guo [157]; (b) Navigational path traced by Manifold ANFIS approach

(b) A dynamic neural network has been proposed by Li and Guo [157] to solve the AUV path planning problem in 3-D complex underwater environments without currents, with

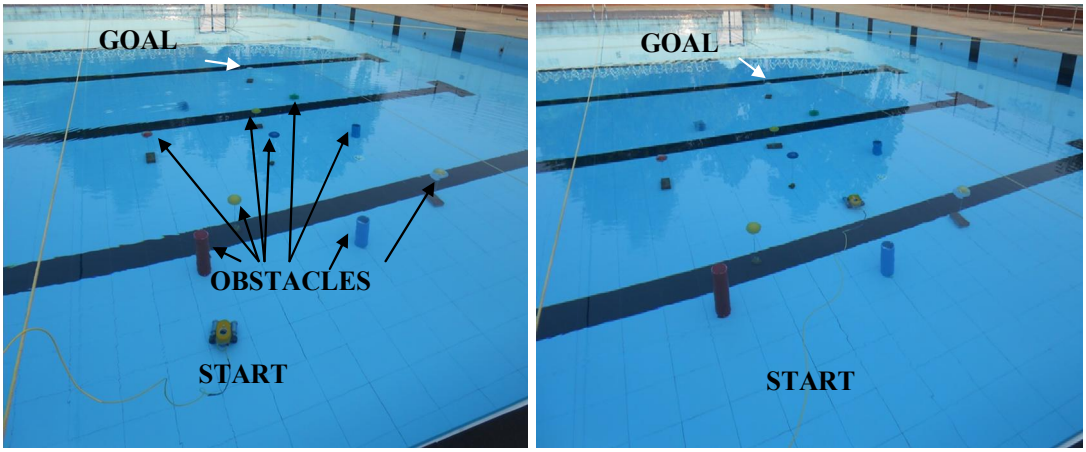
constant currents and also with variable currents. A potential field approach has been implemented among neurons of the network to derive optimal path towards goal avoiding obstacles within the workspace. For comparison, simulation result without considering currents has only considered. Figure 4.13(a) shows that in 3D environment robot moves around a peak of the uneven basin and able to reach goal avoiding obstacles. The proposed algorithm has been applied in simulated environment which is almost same as Figure 4.13(a). CPU time used here is 5.53 sec which less than 6.25 sec as given by Li and Guo [157]. During navigation (in Figure 4.13(b)) more clearance from obstacles has been observed than the previous result (Figure 4.13(a)). The minimization in path length (in terms of cm) has also been notified in Table 4.6 comparison with measured path length of simulation result of Figure 4.13(a).

Table 4.6: Comparison between Proposed Manifold ANFIS based Navigation Strategy and other navigational strategies for three-dimensional simulated environments

Figure No.	Navigational Strategy	Length of 3D path traced by underwater robot (in cm)	Deviation (in %)
4.12(a)	Collision-free approach [156]	794.8	2.38
4.12(b)	Proposed ANFIS based Navigation	776.3	
4.13(a)	NN based Approach [157]	1059.78	1.18
4.13(b)	Proposed ANFIS based Navigation	1047.42	

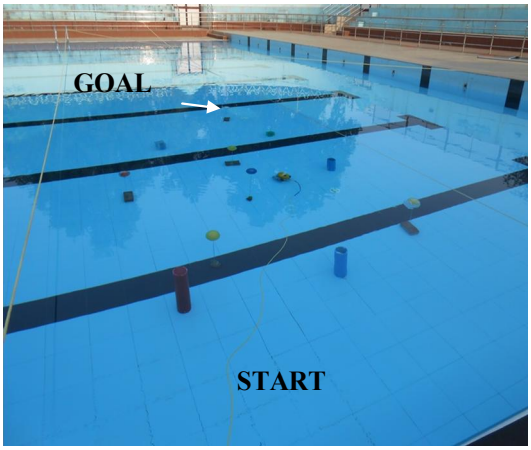
4.6 Experimental Results and Comparisons

Real time experiment has been organized using GNOM Baby underwater robot for validation of simulation results based on Manifold ANFIS approach. The specifications of the small size underwater robot have been incorporated in Appendix-A. The data of depth sensor and ultrasonic sensors can be stored as well as monitored in online manner from workstation. The MATLAB code used in simulation mode has been interfaced with Arduino microcontroller of "GNOM-Baby" to achieve desirable performance in real world underwater environment. The navigational performance of proposed path planning algorithm during simulation study has been justified by performing real-time experiment of underwater robot GNOM-baby in Figure 4.14 which has obstacle arrangement similar as simulation scenario of Figure 4.8.

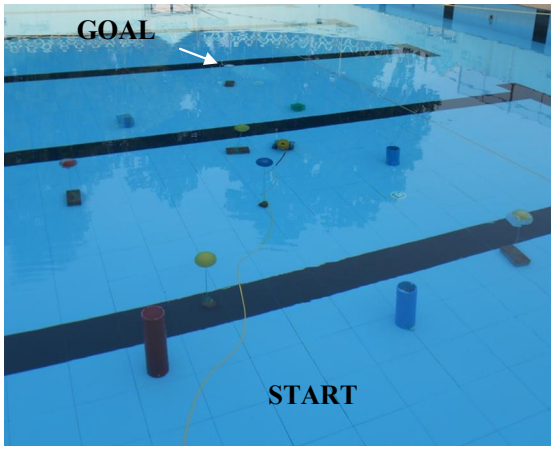


Stage: 1 (Initial Position)

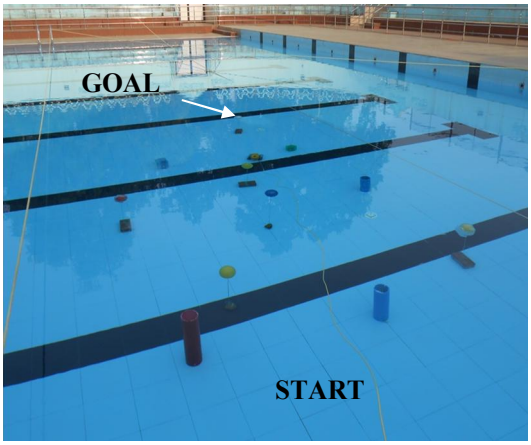
Stage: 2



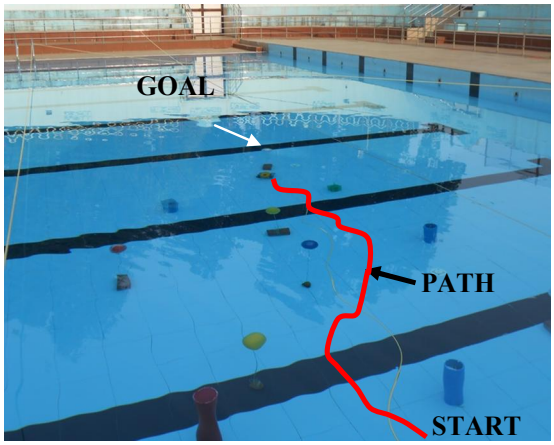
Stage: 3



Stage: 4



Stage: 5



Stage: 6 (Final Position)

Figure 4.15: Experimental views for GNOM Baby embedded with Manifold ANFIS approach based navigational Strategy for scenario same as Figure 4.8

The path obtained in real time environment (Figure 4.14) has been found to be approximately similar with the path followed by ANFIS based navigational strategy during the simulation study (Figure 4.8). The performance of the proposed navigation methodology has been assessed on the basis of path length (in 'cm') and time (in 'sec') taken by underwater robot to reach target, which is indicated in Table 4.7. Deviation between simulated and experimental performances has been recorded within reasonable limit.

Table 4.7: Comparison between experimental and simulated studies on three-dimensional navigation

Type of Environment	Path length (in cm)	Error (in %)	Time taken (in sec)	Error (in %)
Simulated (Figure 4.8)	1065.6	6.5	28.72	6.79
Experimental (Figure 4.14)	1134.9		30.67	

4.7 Summary

The reasoning ability and self-learning mechanism of ANFIS approach has been implemented here to deal with the uncertainties and nonlinearities involved in underwater motion. Major findings of executed investigations on ANFIS based reactive navigation of underwater robot have been outlined here:

- ANFIS models have been employed in a manifold manner to determine the required change in heading angle of underwater robot in horizontal as well as vertical plane for avoiding obstacles.

- Back-propagation based training of ANFIS models have been performed with respect to a large number of training and testing patterns which have been chosen empirically.

- Several simulations have been carried out to search suitable values of weight parameters (K_{oa} and K_{ts}) which have taken part in fusion of ANFIS models. By assessing path lengths in simulated environment on trial and error basis, fine-tuning of weight parameters has been achieved approximately.

- Fusion of reactive behaviours (such as obstacle avoidance, wall following etc.) may avoid traps like local minima situation, but the path may not be globally optimal.

- Comparison of proposed approach with previous researches on underwater navigation certifies feasibility of algorithm.

- Simulation results have been validated by performing experiments within real world underwater environment. Simulated and experimental performances have been compared with each other in terms of path length and travel time. Average error between simulated and experimental results for Proposed Manifold ANFIS approach based navigational strategy is found to be within 7% (6.5% for path length and 6.79% for travel time).

Satisfactory performances in simulation and experimental mode may endorse the effectiveness of proposed behavioural architecture based on ANFIS models. Navigational approaches based on evolutionary based optimization algorithms have been addressed and analysed in subsequent chapters.

5. Analysis of Adaptive Shuffled Frog Leaping Algorithm for Underwater Robot Navigation

Path optimization through neuro-fuzzy reasoning solely depends on human perception and expertise about the navigational scenario and robotic behaviors as seen in previous chapter. For achieving higher accuracy, a fresh attempt has been made here to implement evolutionary algorithm (EA) based motion control for underwater robot which has already become an emerging field for research as discussed in Chapter 2. Based on evolution strategy, EAs can be categorized as genetic and memetic algorithms. By mimicking efficient social behavior of species, memetic algorithm has successfully provided fast and robust solution to complex nonlinear optimization problem with much less computational complexity than genetic algorithms which represents biological evolution [109]. One of the newest forms of memetic algorithm has been considered here to be implemented as navigational strategy for three-dimensional motion.

5.1 Introduction

The concern of present research work is to find out the robust and accurate navigational methodology for underwater robot which is an extremely challenging on-going issue. Two basic objectives of any path planning problem, minimization of path length and obstacle avoidance can be treated as optimization issues of evolutionary based approaches. Memetic algorithms (MAs) are a distinctive class of evolutionary computing methods which perform parallel local searches by involving all individuals of population to move towards the global optimum. Shuffled Frog Leaping Algorithm (SFLA), as developed by Eusuff and Lansey [108], is a meta-heuristic optimization method that models the food searching behavior of a group of frogs in terms of memetic evolution. SFLA combines the concepts of PSO like deterministic search and randomization based on Shuffled Complex Evolution (SCE) [109]. Elbeltagi et al. [158] have found that SFLA retains faster convergence speed, easier implementation, fewer parameters, higher success rate and better search quality than other EAs like GA, PSO, and ACO. The SFLA has already been effectively implemented to reveal the global optimal solution for several combinatorial optimization problems. Like other EAs, SFLA also suffers from premature convergence due to lack of diversity within the search space [159]. Though several modifications on original version of SFLA have already been proposed by researchers

[160-163] to avoid such drawbacks, still their performances has not been universally verified irrespective of dimension and nonlinear behaviors of real world problems. In this chapter, new adaptive version of SFLA has been designed and implemented for underwater path planning to provide a fair trade-off between wide and deep search. Convergence behaviour and path optimization ability of proposed Adaptive SFLA based navigational strategy has been evaluated here by executing simulation and experimental results within impulsive three-dimensional environments.

5.2 Basic Mechanism of Shuffled Frog Leaping Algorithm

The Shuffled Frog Leaping Algorithm considers a population of virtual frogs (every frog represents a possible solution) of same configuration but with different adaptation abilities which can individually leap within a swamp or search space [108]. A number of stones containing food are assumed to be distributed within the swamp in a discrete manner. The virtual frogs are motivated to leap towards the stone associated with the highest amount of food (optimal position) as soon as possible. Virtual frog's adaptive ability can be referred as its fitness in SFLA based optimization problem. The virtual frogs symbolize memetic vectors which can interact with each other and also improve their fitness by exchanging memes or information. The fitness of individual has been considered as key factor of memetic evolution to decide whether the virtual frog will jump to the next position or not. Meme represents the individual's idea or behaviour or culture which can be transferred from one to another without performing any genetic operations [109]. Quality of meme or fitness can be improved by controlling leaping direction and step size for each virtual frog.

SFLA replicates the food the finding process of frogs as a stochastic optimization method through three major steps: Partitioning of population, Local search or memetic evolution within each sub-population and shuffling of them within search space as visualized in Figure 5.1. After random initialization, population of virtual frogs is normally divided into parallel sub groups (memplexes) based on their cultures or type of memes. An example of 24 virtual frogs which are equally partitioned into four memeplexes has shown in Figure 5.1(a). Here, variation in quality of memes has been denoted by frogs of different colours. To avoid local minima situation during search, memplexes are also divided into sub-memplexes as shown in Figure 5.1(a). Memetic evolutions of virtual frogs are independently performed within each memplex for all probable directions of

search space. In each iteration of memetic evolution, the position of worst fitted frog in a memplex has been updated with respect to the local best position of the same memplex as follows:

$$X_{new}^{i+1} = X_{worst}^{i} + rand()(X_{LB}^{i} - X_{worst}^{i}) \tag{5.1}$$

Where, X_{worst}^{i}: Worst fitted virtual frog of recent memplex in ith iteration of local search; X_{LB}^{i}: Local best position in current memplex for ith iteration; X_{new}^{i+1}: Updated position of worst fitted virtual frog to be included in next iteration; rand( ): Random value between 0 and 1.

If the fitness of newly updated virtual frog is not better than the worst fitness of the considered memplex then X_{LB}^{i} will be replaced by X_{GB}^{i} (Global best virtual frog of search space) in eq. 5.1. In adverse cases, if the fitness of new virtual frog has not been enough improved, then one randomly generated virtual frog will be included within the memplex by replacing the worst fitted virtual frog.

Figure 5.1(b) shows the locations of the frogs after few iterations of local search process. As iteration progresses, all virtual frogs of a memplex will gradually leap towards the best fitted position of that memplex [164]. After a specified number of evolutions, all memplexes are shuffled together to enhance the quality of memes in population through global exploration of virtual frogs with different cultures. Shuffling process enhances the diversity of population irrespective of any partiality. After updating global optimal solution of population, the shuffled population will be redistributed into memplexes to perform local search again. After few cycles of partitioning and shuffling processes in a sequential manner, positions of virtual frogs within four memeplexes which are nearer to the locations with higher amount of food have been shown in Figure 5.1(c). The evolution and shuffling processes will continue alternatively for several times until all virtual frogs reach the global best position. Both local and global searches are performed here in a cooperative manner to ensure the flexibility and robustness of optimization algorithm [113]. Apart from fast convergence speed, SFLA has found to be successful in local minima avoidance. The objective of this memetic based search process is to find out near optimum solution for large scale complex optimization problems which are difficult to be solved by gradient based mathematical approach.

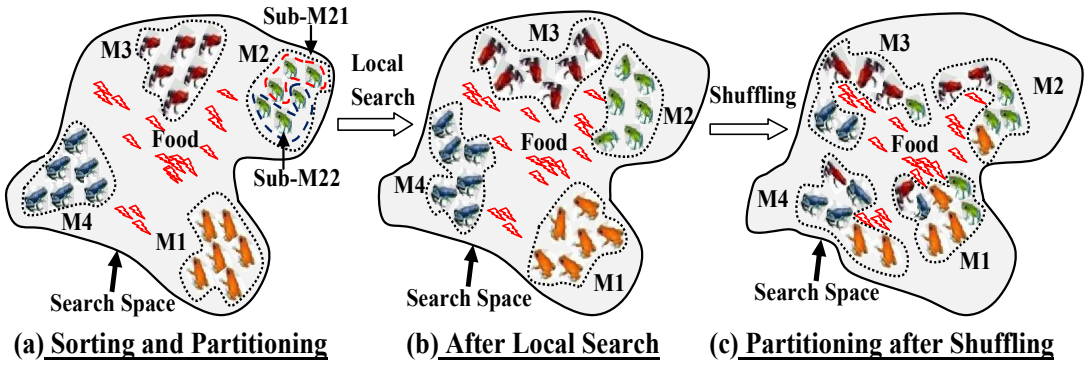


Figure 5.1: Change in positions of virtual frogs within search space at different stages of SFLA: (a) Initial state of virtual frogs within population which is partitioned into memplexes and submemplexes; (b) Positions of virtual frogs within each memeplex after few iterations of local search; (c) Positions of each frog within each memeplex after few cycles of memetic evolution and shuffling

In memetic evolution strategy of conventional SFLA (as shown in eq. 5.1), the leaping step size for worst fitted virtual frog has been significantly affected by the position of local best or global best frog of current iteration. So, the probable new position of updated frog will always be limited by a line segment between current worst position and the local or global best position of virtual frog as shown in Figure 5.2 (a) [160]. By following the described leaping rule, virtual frogs of a memplex will never reach the neighbourhood region which owns more food than the current best position. Therefore, in spite of good convergence speed, conventional SFLA may not provide the solution with best quality [161]. Such limitation may degrade the optimization ability which may result in premature convergence or local minima problem.

Moreover, imperfect perceptions about search space may fail to locate the best fitted location precisely. In that case, estimation of leaping step size for worst virtual frog may result in erroneous solution. So, the frog leaping rule as shown in Figure 5.2(a) may not accurately represent the vagueness inherited in real world situation. In other way, consecutive iterations of local search by following memetic evolution strategy of eq. 5.1 may produce large number of virtual frogs with similar fitness value which may result in loss of diversity within population. So, the optimization process may be stagnated before reaching global optima during next iterations.

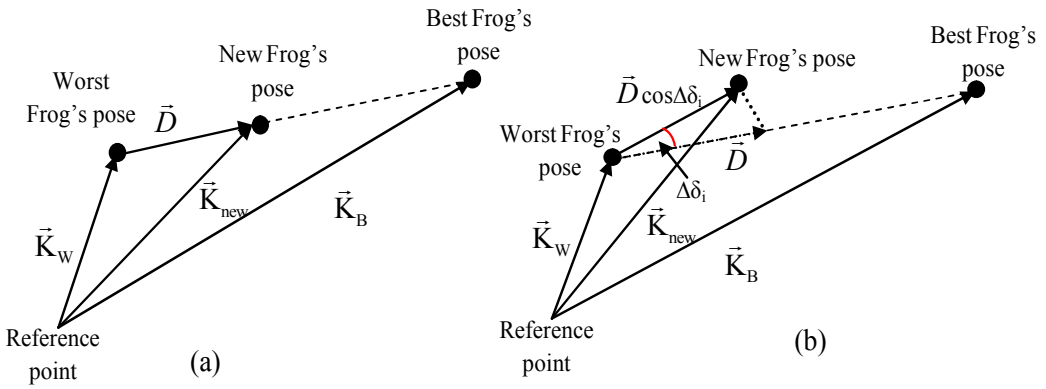


Figure 5.2: Vector representation of perturbation rule for worst fitted virtual frog's pose in (a) conventional SFLA and (b) proposed Adaptive SFLA

5. 3 Adaptive version of Shuffled Frog Leaping Algorithm

In conventional SFLA, leaping step size has been scaled using a random value (in eq. 5.1) which may not ensure the required amount of change in virtual frog's position. So, despite of faster convergence speed, SFLA may suffer from imprecise solution, lack of diversity, lower search quality, local minima problem etc. [162]. Several modified versions of SFLA have already been originated to improve SFLA's performance while solving nonlinear complex optimization problems [165-167]. In present research work, a novel frog leaping rule for improving SFLA's performance has been proposed by multiplying an adaptive term with leaping step size of worst fitted virtual frog which is also associated with inertia component of previous leaping step size. Leaping direction of virtual frog has been controlled in proposed evolution strategy. Apart from adaptive local search within memplex, new rule has also been proposed for updating global optimal position during shuffling iterations. The objective of the proposed adaptive version of SFLA is to perfectly combine the benefits of both local and global searches. Pseudo code for SFLA along with proposed adaptive frog leaping strategy has been depicted below:

Pseudo Code for Adaptive Shuffled Frog Leaping Algorithm:

Step 0: *Initialization of control parameters*

m: number of memeplexes;

r: number of solution vectors or virtual frogs in each memeplex;

Total population size, P= mr

Maximum iteration number or shuffling iterations (is_{max});

Number of decision variables in each vector = d;

Boundaries for decision variables: Lower bound (LB^d) and upper bound (UB^d)

Step 1: *Generate a virtual population*

Virtual Frog Population: {k(1), k(2),…….., k(P)};

v^{th} virtual frog can be represented as, $k(v)=\{k_v^1, k_v^2, ..., k_v^d\}$;

Initialize virtual frogs in population:

For v=1: P

For j=1: d

$$k_v^j = LB_v^j + rand(0,1)\bullet(UB_v^j - LB_v^j) \quad (5.2)$$

End for

End for

Global Search Process:

After completion of initialization process, iterations of global optimization process (is) will start at step 2 and continue up to specified is_{max} or until stopping criterion is satisfied.

Step 2: *Fitness Evaluation and Sorting*

For v=1: P

Find out f (v) =fitness of k (v) (5.3)

End for

Sort the set of virtual frogs in order of decreasing fitness value and store them in an array, K (is) = {k(v), f (v); v = 1, . . . , P };

Record the best fitted vector as, K_G (is) = k(1) (5.4)

Step 3: *Partition frogs into memeplexes*

Partition of array K into 'm' memplexes which can be denoted as M (ih) (where, ih: 1, 2,…., m). Each memplex is assumed to contain 'r' vectors. The distribution of population within memplexes has given in Figure 5.3. Round robin fashion has been followed as partitioning method i.e., the first vector k(1) from sorted array of vectors 'K' will be placed in first memplex denoted as M(1). Sequentially, k(2) will move to second memplex M(2) and the process will continue up to k(m) which will be located at last memplex M(m). Number of memplexes 'm' is considered to be much less than population size 'P' (m<<P). Next, vector k (m+1) will be stored again in first memplex M(1), k(m+2) in M(2) and so on. Finally, the last vector k (P) will be kept at memplex M(m) as shown in Figure 5.3.Such distribution process of virtual frogs facilitates to maintain the diversity in population [109].

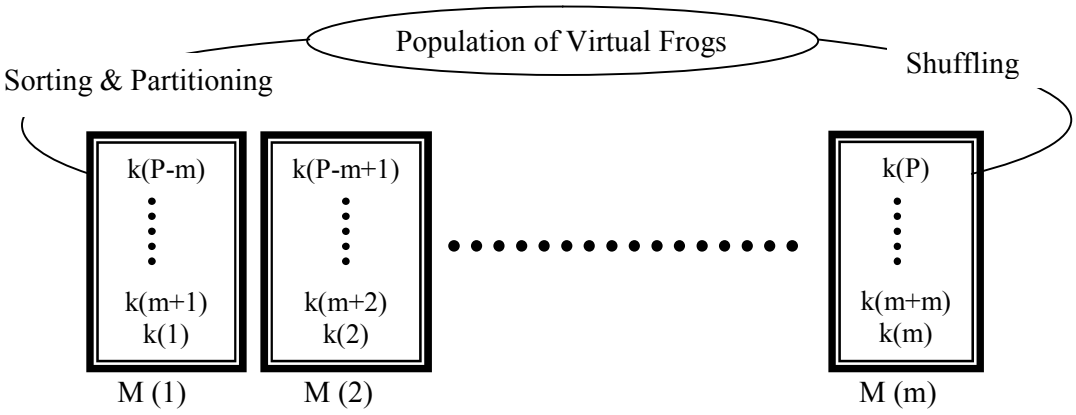


Figure 5.3: Distribution of Memplexes

Step 4: *Memetic evolution within each memeplex*

Evolve each memeplex as per the ***Local Search*** procedure of SFLA (as narrated below)

Step 5: *Shuffling of memeplexes*

After a defined number of memetic evolutionary steps within each memeplex, replace all memplexes into K(is), such that $K(is) = \{M(1), ..., M(m)\}$;

Again sort array K in order of decreasing fitness value and Update the global best solution vector (K_G) of population which will be used in next iteration. New adaptive rule

has also been proposed for updating global best position,

$$K_G(is+1) = K_G(is) + \frac{is_{\max}}{is} \bullet (fitness(K_G) - fitness(K_W)) \qquad (5.5)$$

Where, is: Counts of shuffling, is_{max}: Maximum no. of shuffling iterations

Step 6: *Check convergence*

If the iteration has reached its maximum value is_{max}, then computation may come to an end. In other way, if there is negligible difference between finesses of global best vectors of two consecutive iterations, then iterations will be stopped and best fitted vector may be considered as near optimal solution. Otherwise, return to step 2.

Local Search Process:

Step 4a: *Initialization of variables for local search*

'iq': Counts for the number of submemeplexes (1,2,…..q)

'i*l*': Counts for the number virtual frogs within a submemeplex (1,2,…..*l*)

'im': Number of evolution within a memplex (1 to im^{max})

Step 4b: *Build a submemeplex*

Here, '*l*' distinct frogs are selected randomly from each memeplex to build a submemplex. 'q' no. of submemplexes are assumed to be formed per memplex i.e. r = *l*q. Each submemeplex has also been sorted in descending order of their fitness value, such as best fitted virtual frog in iq^{th} submemplex is denoted by $K_{B,\,iq} = k_{iq}(1)$ {where 'i*l*'=1 and $K_{B,\,iq} = k^{j}_{B,iq}$; j=1,…,d)} and worst solution vector is referred as $K_{W,\,iq} = k_{iq}(l)$ {where 'i*l*'=*l* and $K_{W,\,iq} = k^{j}_{W,iq}$; j=1,…,d)}

Step 4c: *Adaptive evolution of worst fitted vector*

For ih=1: m

For im=1: im^{max}

For iq =1: q

For j =1: d

$$S(im) = w_{im} S(im-1) + \frac{fitness(K_{W,iq})}{fitness(K_{B,iq})} \bullet \frac{im^{\max}}{im} (k^j_{B,iq} - k^j_{W,iq}) \bullet \cos(\Delta\delta_{im}) \qquad (5.6)$$

If $S_{min} < S\ (im) < S_{max}$

$$k^j_{new,iq} = k^j_{W,iq} + S(im) \qquad (5.7)$$

Endif

End for

If $fitness(K_{new,iq}) < fitness(K_{W,iq})$

Replace $K_{W,iq}$ of specified submemplex by $K_{new,iq}$ (5.8)

Else

Replace local best, $K_{B,iq}$ by global best vector of population, K_G in eq. 5.6 and perform the tuning operation to get, $K_{new,iq}$.

If $fitness(K_{new,iq}) < fitness(K_{W,iq})$

Replace $K_{W,iq}$ of specified submemplex by $K_{new,iq}$ (5.9)

Else

Generate new population in a random manner:

For j=1: d

$$k^j_{rand,iq} = LB^j + rand(0,1) \bullet (UB^j - LB^j) \qquad (5.10)$$

End for

Replace $K_{W,iq}$ by $K_{rand,iq}$ (5.11)

Endif

Endif

End for

End for

End for

Step 4d: *Upgrade the memplex*

After completion of replacement process for worst vector or virtual frog in each submemplex, put the array of all submemplexes in their original memplexes which are also returned to the population array in step 5.

SFLA's performance solely depends on how accurately the leaping step size has been determined in each evolution step of memplexes. So, extension or reduction of leaping step size during memetic evolution has been proposed in an adaptive manner in eq. 5.6. Here, objective is to achieve faster convergence along with higher search quality. As fitness of virtual frog is the major evaluation index of any optimization process, so, the ratio of worst ($f(K_{W,iq})$) and best ($f(K_{B,iq})$) finesses of current iteration has been included as an adaptive scaling factor of the leaping step size in local evolution strategy of eq. 5.7. Ratio of maximum and current iteration number has also been multiplied with the new scaling factor to make the proposed strategy more adaptive by nature. The proposed adaptive term may enhance or reduce the leaping step size based on updated value of fitness of population and current iteration number. At initial stage of search, as $f(K_{B,iq}) << f(K_{W,iq})$, large value of scaling factor will enhance the leaping step size which will facilitate global exploration of search space. With the progress of iterations, worst virtual frog's position will be closer to local best position. So, fitness of worst virtual frog will be improved $f(K_{B,iq}) \approx f(K_{W,iq})$ and the leaping step size will be scaled down to provide a small change in worst frog's position. Variation in iteration number from low to high (im^{max}) in eq. 5.6 will also show the effect on modification of leaping step size same as fitness ratio. Therefore, finite value of proposed adaptive acceleration factor can regulate the convergence speed in various stages of SFLA based optimization. To overcome the limitation of SFLA as described in Figure 5.2 (a), the leaping direction of worst virtual frog has been controlled here by introducing a variable deviation angle ($\Delta\delta_{im}$) in frog leaping strategy.

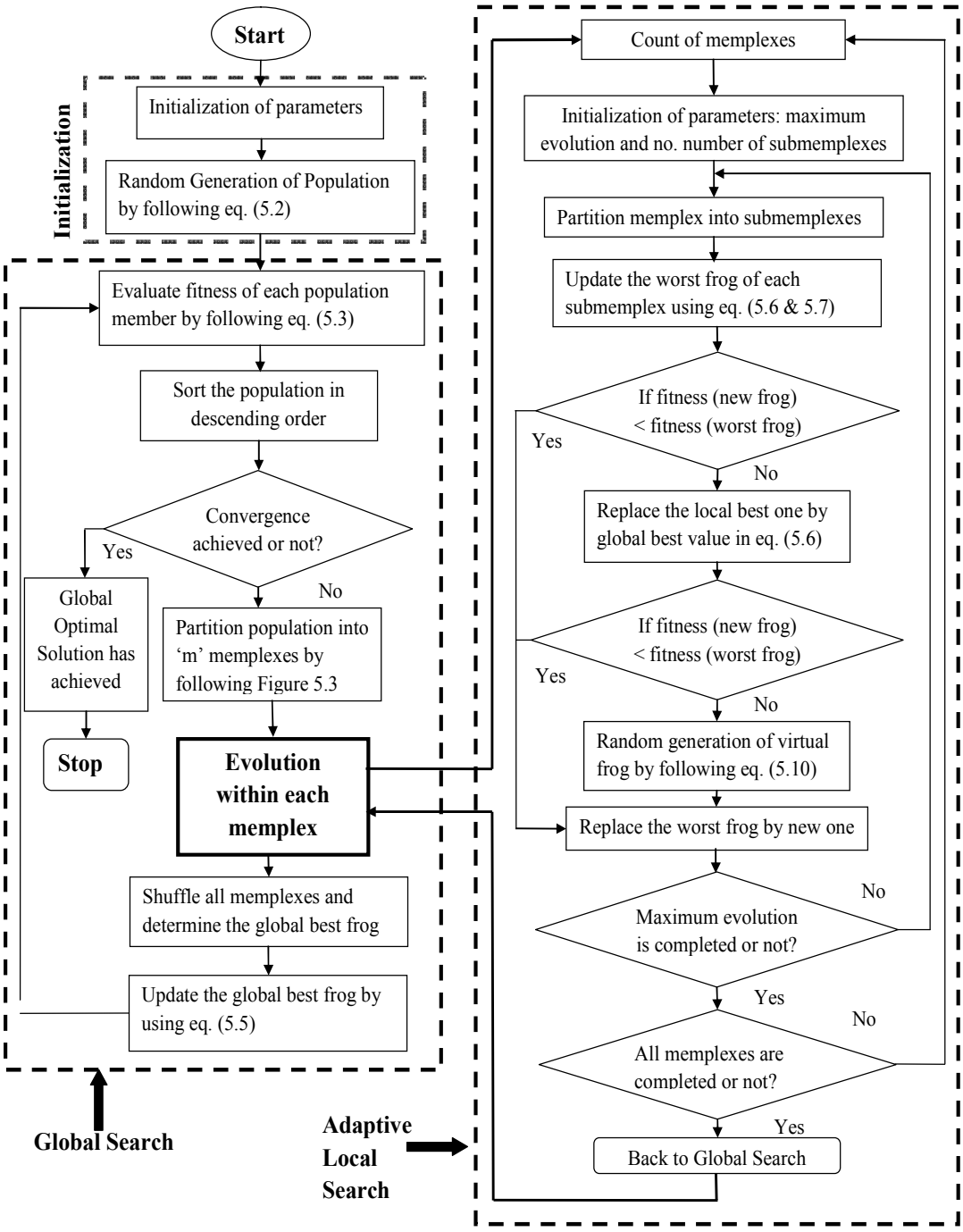


Figure 5.4: Flowchart for proposed adaptive version of Shuffled Frog-Leaping Algorithm

Updated position of worst fitted frog by following proposed adaptive frog leaping rule (eq. 5.6) has been shown in Figure 5.2(b). Motion direction of worst fitted frog has been regulated within the specified range {(0 to $\pi/2$) or ($-\pi/2$ to 0)}. The variation of $\Delta\delta_{im}$ with respect to iteration number can be defined as follows:

$$\Delta\delta_{im} = \frac{\Delta\delta_{\max}}{im} \tag{5.12}$$

Where, $\Delta\delta_{im}$ denotes the angle of deviation from virtual frog's original direction of motion from worst to best positions of memplex in the im^{th} iteration of local search; $\Delta\delta_{max}$ specifies initial maximum value of deviation angle.

At the initial stage of algorithm, leaping direction of worst fitted virtual frog has been proposed to differ from the line which can connect the worst and best locations within memplex by maximum deviation angle $\Delta\delta_{max}$ as per eq. 5.12. Therefore, maximum widening of global search space can be achieved primarily as worst virtual frog has been enabled here to leap beyond the determined local best position of memplex. With the increase of iterations, worst virtual frog will start to shift towards the best fitted location. In that case, deviation angle will be progressively decayed from its maximum value by following eq. 5.12 to enhance the local exploitation ability rather than explorative nature of algorithm. At final stage of iterations, deviation in leaping direction of worst fitted virtual frog will be negligible. Therefore, variations in worst virtual frog's motion direction based on iteration can be beneficial for maintaining a smooth balance between diversification and intensification in search process.

A specific observation on original evolution strategy for each memplex (eq. 5.1) exposes that collective behaviour of frogs for leaping towards the higher amount food effectively depends on the inertia of frogs which denotes the individual's nature to stay fixed in its current position. So, inertia effect of worst frog's previous position has been counted in the proposed frog leaping rule of eq. 5.6 to enhance degree of accuracy while modifying the worst fitted solution vector. In proposed evolution strategy, leaping step size of current iteration (S (im)) will depend on the worst frog's leaping step size of previous iteration (S (im-1)). Inertia factor will be kept as high in earlier iteration to explore the search space through large leaping step size of worst virtual frog. As iteration progresses, this perceptual inertia value is required to be decreased in a linear manner for fine tuning of worst member. So, linear variation of inertia factor relative to iteration number can be formulated as follows:

$$w_{im} = w_{\min} + \frac{im^{\max} - im}{im^{\max}} \bullet (w_{\max} - w_{\min}) \tag{5.13}$$

Where, w_i: inertia factor of i^{th} iteration; w_{max}: highest value of inertia; w_{min}: lowest value of inertia; im^{max}: Maximum evolution number of a memplex.

Inclusion of variable inertia factor in frog leaping method has shown a significant impact on the convergence behaviour of the algorithm by controlling global exploration and local exploitation in a cooperative manner.

Moreover, in conventional SFLA, global best position for virtual frog may not be changed in successive iterations as other virtual frogs of population will always be less fitted than global optimal virtual frog. Due to variation in search space, at a certain moment, more amount of food may found in a location which is different from the previously defined global best position. In that case, global best position must be updated in an online manner to maintain the accuracy in optimization process. So, apart from new adaptive local search strategy, separate rule for updating global best position for next iteration has also been proposed and successfully implemented in eq. 5.5. This new rule has found to be advantageous for enhancing both convergence speed and diversity of population up to some extent. Therefore, possibility of being trapped in local minima situation may get reduced.

5.4 Implementation of Adaptive SFLA as Path Planning Algorithm:

For a certain pose of underwater robot during navigation, if on-board sensors detect any obstacle within their specified range, Adaptive SFLA (as per flowchart given in Figure 5.4) will be actuated to estimate robot's next best pose which must not collide with obstacle. Virtual frogs in population of SFLA denote the feasible solutions of given problems which are randomly generated within specified boundary. Therefore, population of proposed SFLA based path planning algorithm will be randomly initialized by all possible next poses of underwater robot $\{v_p{:}v_{x,p},v_{y,p},v_{z,p};p=1,2,...P\}$ as visualized in Figure 5.5. The coordinate of any population vector must be bounded within the robot's current k^{th} pose of robot $\{robx_k,roby_k,robz_k\}$ and the nearest obstacle's coordinate $\{nobx,noby,nobz\}$ as follows:

$$\begin{aligned} &If \quad robx_k < nobx < goalx; \quad then \quad robx_k \le v_{x,p,i} \le (nobx - r_o); \\ &or, If \quad goalx < nobx < robx_k; \quad then \quad (nobx + r_o) \le v_{x,p,i} \le robx_k; \end{aligned} \tag{5.14}$$

Where, *i*= iteration number of path planning algorithm; p= number of members within population; r_o = clearance from centre coordinate of nearest obstacle which must be slightly more than highest dimension of respective obstacle as shown in Figure 5.5.

Limits for y and z coordinate of population members will also follow the above mentioned rules in their respective way. So, random initialization can be formulated in a generalized manner as follows:

$$v_{j,i,p} = rob_{j,k} + rand(0,1)\bullet\{nob_j - rob_{j,k}\} \tag{5.15}$$

Where, j= (x, y, z): Dimension of positional vector

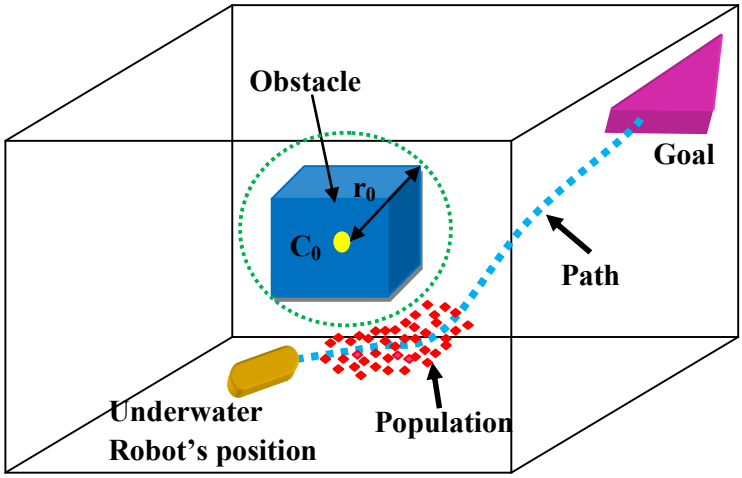


Figure 5.5 Initialization of population on obstacle detection

Pseudo code of adaptive SFLA has signified that initial values of control parameters are of high importance to control the optimization efficiency of proposed navigational strategy. As per reviews conducted, researchers have used different values for SFLA's parameters based on the specifications of real world problems to be optimized. It can be stated that there is no conventional rule to decide suitable values of control parameters in SFLA based optimization problem. A specific range of parameters for proposed adaptive SFLA have been considered here in Table 5.1 to perform three-dimensional path optimization within MATLAB based simulated scenario. The influences of control parameters on adaptive SFLA's performance have been analysed here in detail:

Due to the stochastic nature of SFLA, accuracy of finding global optima may improve with the increase in population size. But large population size may adversely enhance the

computational burden of optimization problem. So, a proper trade-off between reduction in computational complexities and higher degree of optimality in solution of considered optimization problem must be maintained. Suitable range of population size (30 to 300) for solving current research problem has been mentioned in Table 5.1.

Both number of memplexes and uniform distribution of virtual frogs in it are of great importance for improving convergence speed and probability of getting near optimal solution in SFLA based optimization. Memplex with less number of virtual frogs may nullify the advantage of memetic evolution based local search process. In other way, large number of virtual frogs in memplex may also enhance the computational overheads. Submemplex with very less number of virtual frogs may reduce the convergence speed resulting in lengthier run time. In reverse case, submemplex with large number of virtual frogs may raise the probability of deviation in direction the local search which may result in random generation of new virtual frog to replace the worst virtual frog. The values of control parameters for proposed adaptive SFLA based navigational strategy have been specified in Table 5.1.

Table 5.1: Parameters for Adaptive SFLA based navigational strategy

Parameters	Range of parameters
Population Size	30-300
Dimension of population member	3
No. of memplexes and size of memplex	5-10, 6-30
No. of submemplexes and size of submemplex	2-6, 3-5
No. of evolutions within each memplex	10-30
No. of shuffling of memplexes or global iterations	20-50
Range for inertia parameter (w_i)	$w_{max} = 0.9$ and $w_{min}=0.4$
Maximum value of deviation angle	$\Delta\delta_{max} = \pi/2$
Maximum and Minimum Step size	$S_{max} = 0.93$and $S_{min} =0.01$

The maximum number of evolution within each memplex, im^{max} also influences the searching speed of SFLA based optimization. For low value of im^{max}, shuffling of memplexes will occur in a frequent manner resulting poor local search ability. On the other hand, for large value of im^{max}, search within each memplex may get stuck in local optima. Avoiding all extreme conditions, the maximum number of evolutions within each

memplex has been varied for present research problem based on size of memplexes as given in Table 5.1. Quality of global search for SFLA based optimization depends on how many times the memplexes are shuffled and repartitioned into memplexes. But after certain times of shuffling of memplexes, it may not influence the search quality of solutions and there may be unnecessary increase in computational time.

Favourable number of shuffling iterations has been selected based on size of population and its memplex numbers as shown in Table 5.1. Another important parameter of SFLA based navigation is the maximum value of step size which provides maximum permissible change in the worst virtual frog's position to improve its fitness in each evolution of memplex. Basically, S_{max} and S_{min} are the constraints for controlling the global exploration and local exploitation ability of SFLA in a coupled manner. Small values of step size may reduce the diversity of local search process and reversely, large step size may fail to detect the exact optima due to lack of fine tuning ability. Therefore, specific values of S_{max} and S_{min} for current research work (as given in Table 5.1) have been decided by trial and error analysis.

After initializing all the required parameters, the steps of Pseudo Code for Adaptive SFLA have been followed (as described in Section 5.3) to find out next possible pose of robot. Before completion of maximum memetic evolutions within memplexes or maximum times of shuffling of memplexes, if the fitness of global best solution within the population has reached its specified minimum value or changes in the fitness of best solutions in several consecutive iterations are negligible, then iterations of adaptive SFLA will be stopped (as given in flowchart of Figure 5.4). The best fitted solution among current population set will be considered as next near optimal position for underwater robot $\{robx_b, roby_b, robz_b\}$. To reach the best pose, underwater robot has to take a proper turn. The deviation in heading angle due to presence of obstacle will be estimated by angular difference between current $\{robx_k, roby_k, robz_k\}$ and next best pose of robot $\{robx_b, roby_b, robz_b\}$ as follows: $\Delta\phi_{b,k+1} = \tan^{-1}\left(\frac{robz_b - robz_k}{\sqrt{(robx_b - robx_k)^2 + (roby_b - roby_k)^2}}\right)$

and $\Delta\psi_{b,k+1} = \tan^{-1}\left(\frac{roby_b - roby_k}{robx_b - robx_k}\right)$ (5.16)

for vertical and horizontal plane respectively.

Therefore, exact target angle of robot to reach the next position that robot will actually follow can be defined as follows:

$$\phi_{actual,k} = \phi_g + \rho_v \Delta\phi_{b,k+1} \text{ or } \psi_{actual,k} = \psi_g + \rho_h \Delta\psi_{b,k+1} \quad (5.17)$$

Where, ρ_v and ρ_h are constant and their values have been chosen empirically.

The deviation in target angle due to presence of obstacles will be within range of (-45^0, 45^0) for $\Delta\phi_{b,k+1}$ and (-60^0, 60^0) for $\Delta\psi_{b,k+1}$. Here, negative sign in angular difference denotes that robot has to take turn in clockwise and positive sign directs robot to rotate in counter-clockwise direction. Value of $\Delta\phi_{b,j+1}$ or $\Delta\psi_{b,j+1}$ will be zero in no obstacle cases. After crossing the obstacles' region, robot will again align itself towards goal. For any further detection of obstacle, the above described steps will be repeated, until robot reaches the goal as given in flow chart of Figure 5.4.

Evaluation of fitness for each population members can be considered as a performance index for population based metaheuristics in a generalized manner. So, fitness function which is used to evaluate each pose of underwater robot within population has been defined in the following subsection.

5.4.1 Design of Fitness function:

Formulation of fitness function must affect the convergent behaviour of proposed adaptive SFLA based navigational strategy. The sum of squared errors of differences between underwater robot's expected and actual poses or orientations can be projected as fitness function. Such fitness function may face local optima situations for two main reasons:

- The imprecise sensor readings and its range limitation may not distinguish poses of robot.

- The geometrical symmetries in workspace may generate similar sensor readings for different pose of robot which may cause ambiguity in solutions.

Removing all such hurdles, a fitness function for proposed three-dimensional path planning algorithm can be designed as a combination of three main constraints: minimization of path length travelled by robot, safe avoidance of obstacles and reaching target as soon as possible.

(a) Shortening of Path Length: The underwater robot will always try to travel as short as possible to reach its next best fitted position. So, fitness will be directly proportional with the distance between robot and its next pose. It can be represented as follows:

$$Fitness_{p,i} \propto \min_{p \in P} \left\| ls_{p,i} \right\| \tag{5.18}$$

Where, $ls_{p,i} = \sqrt{(vx_{p,i+1} - robx_i)^2 + (vy_{p,i+1} - roby_i)^2 + (vz_{p,i+1} - robz_i)^2}$

$\{\text{robx}_i, \text{roby}_i, \text{robz}_i\}$: i^{th} pose of underwater robot; $\{\text{vx}_{p,i+1}, \text{vy}_{p,i+1}, \text{vz}_{p,i+1}\}$: p^{th} solution vector in population set of probable positions of underwater robot for next iteration.

(b) Obstacle Avoidance: Next best possible position of robot should be at maximum safe distance from nearest obstacle. Fitness will be inversely proportional with the distance between each solution vector and detected nearby obstacle [147], given as follows:

$$Fitness_{p,i} \propto \frac{1}{\min_{p \in P} \left\| NOD_{p,i} \right\|} \tag{5.19}$$

Where, $NOD_{p,i} = \sqrt{(vx_{p,i+1} - nobx)^2 + (vy_{p,i+1} - noby)^2 + (vz_{p,i+1} - nobz)^2}$

$\{\text{nobx}, \text{noby}, \text{nobz}\}$: Obstacle detected as a nearest one for current position of underwater robot.

(c) Goal Seeking: The underwater robot will reach goal as soon as possible if no obstacles are detected by sensors between robot's current pose and target of the workspace [147]. So, the goal seeking behaviour has been incorporated in objective function as follows:

$$Fitness_{p,i} \propto \min_{p \in P} \left\| GD_{p,i} \right\| \tag{5.20}$$

Where, $GD_{p,i} = \sqrt{(goalx - vx_{p,i+1})^2 + (goaly - vy_{p,i+1})^2 + (goalz - vz_{p,i+1})^2}$

$\{\text{goalx}, \text{goaly}, \text{goalz}\}$: Goal position of present scenario.

For each vector, fitness function can be formulated as follows:

$$Fitness_{p,i} = \lambda_{ls} \times \min_{p \in P} \left\| ls_{p,i} \right\| + \lambda_{gd} \times \min_{p \in P} \left\| GD_{p,i} \right\| + \lambda_{nod} \times \frac{1}{\min_{p \in P} \left\| NOD_{p,i} \right\|} \tag{5.21}$$

Where, $\{\lambda_{ls}, \lambda_{gd}$ and $\lambda_{nod}\}$: Scaling constants for enhancement or decrement of amplitudes for the functions associated with them respectively

Three scaling constants will determine which part of fitness function will be prioritized at different stages of search to vary the fitness value. High value of λ_{nod} keeps the robot far away from the obstacles, but for its low value collision may occur. Similarly, for high value of λ_{ls}, the probability of choosing solution vector from population which is nearest to robot's current position will be high. High value of λ_{gd} also denotes the goal is near to robot's next pose ensuring reduction in the path length. For low value of λ_{ls} and λ_{gd}, the traced path may become long. While choosing the values of scaling constants, one constraint must be maintained that the value of λ_{nod} should always be higher than λ_{ls} and λ_{gd} to prioritize obstacle avoidance behaviour of underwater robot than its target seeking behaviour during navigation. Basically, fitness of a next possible position of robot will be reduced when it is nearer to the target and robot's current pose and far away from obstacles. So, minimization of fitness function is mostly desirable. There may be a tendency of choosing low value for scaling constants to achieve faster convergence. Too low fitness value of global best solution may also cause premature convergence. In reverse, higher value of scaling constants may enhance the fitness which supports the possibility of collision. Hence, trial-and-error analysis has been suggested to tune the scaling constants to generate feasible path during navigation.

5.5 Simulation studies for Adaptive SFLA based navigation strategy:

Numerous simulation experiments for underwater motion have been executed in this section to evaluate the navigational performance of proposed adaptive SFLA based path planning scheme. For testing of three-dimensional motion of robot embedded with newly proposed metaheuristic based navigational strategy, different arrangements of static obstacles have been made in between start and goal positions of simulated scenarios. Assumptions for performing three-dimensional navigation within MATLAB based simulated environment have already been discussed in Section 4.5 of Chapter 4. Several simulations have also been enacted to choose the suitable parametric value for adaptive SFLA on a trial and error basis. Convergence behavior and path length minimization ability of proposed adaptive SFLA has also been compared with other evolutionary algorithms for authentication purpose.

5.5.1 Example of robotic behaviours for adaptive SFLA based navigational strategy:

While tracing path from start to goal, three preliminary robotic behaviours (e.g. Obstacle avoidance, Wall following and Target seeking) of underwater robot have been verified for navigational scheme based on proposed adaptive SFLA. In all simulated scenarios (Figures 5.6 - 5.8), safe clearance from obstacles or collision avoidance has been prioritized by newly evolved adaptive SFLA based navigational strategy. Coupling between obstacle avoidance and target seeking behaviours has been observed in Figure 5.6 where only three obstacles are randomly placed in between start and goal points of robot. In obstacle free region, target seeking behaviour has been solely actuated reducing the effect of other behaviours of robot to minimize the path length and travel time of navigation. In simulated path, changes in colour symbolize the activation of reactive behaviours by robot.

To avoid large size obstacle (as shown in Figure 5.7), wall following behaviour of underwater robot has been required to be activated. While performing such behavioural action, robot will make a turn in clockwise or anticlockwise direction to follow a path in parallel with the 'wall' like obstacle. On completion of the influence of wall following behaviour, underwater robot will realign itself in the direction of target. In Figure 5.7, underwater robot has successfully traced path from start to goal without any collision while performing wall following behaviour. In Figure 5.8, the number of obstacles has been increased and their arrangement within the simulated workspace has been done in a chaotic manner. Both obstacle and target seeking behaviours of underwater robot have also been successfully tested for such complex environment. Each simulated experiment has been performed for 20 times and the path with minimum length has been viewed here.

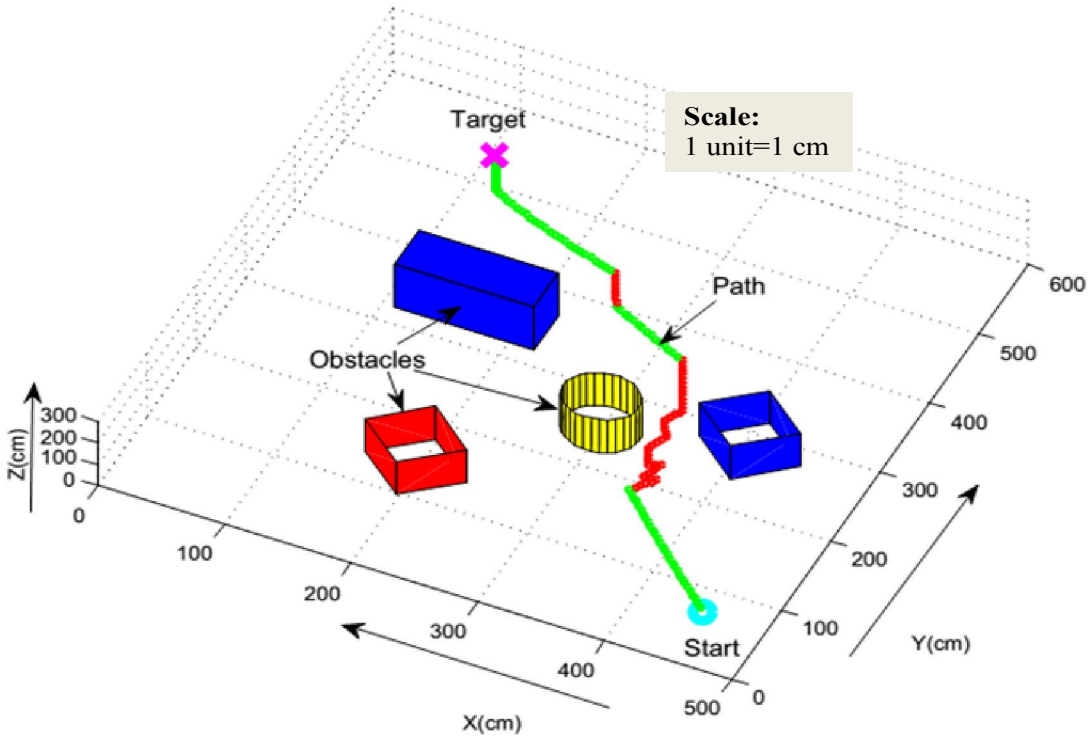


Figure 5.6: Simulation result for reactive behaviours of underwater robot

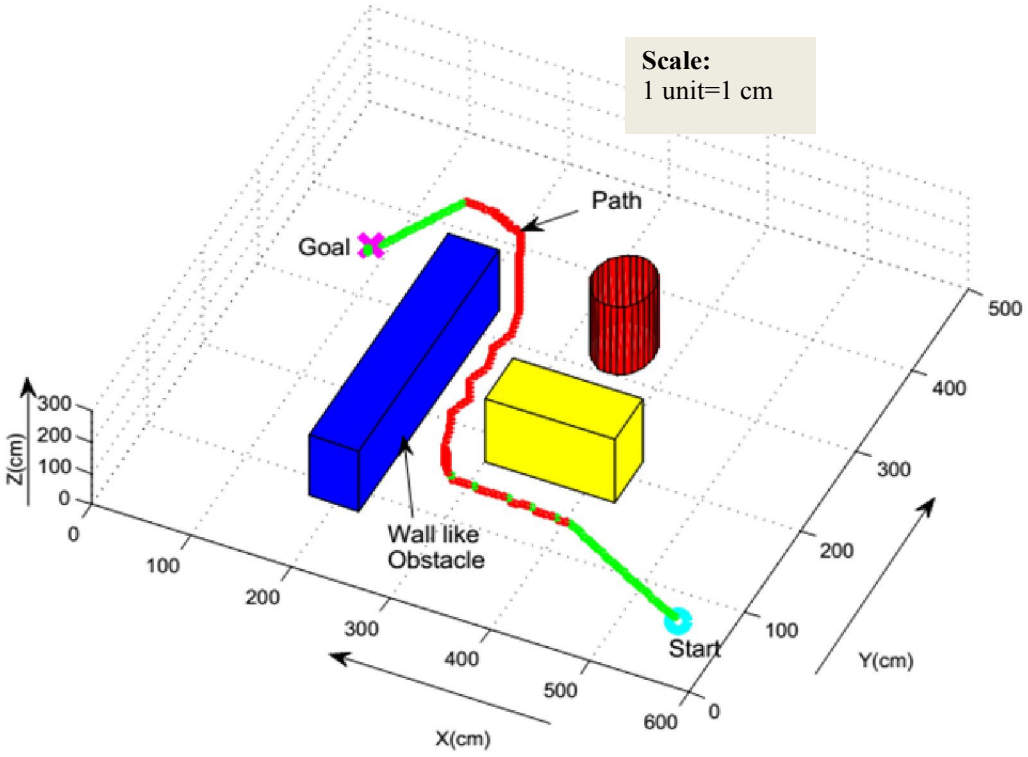


Figure 5.7: Simulation result for wall following behaviour

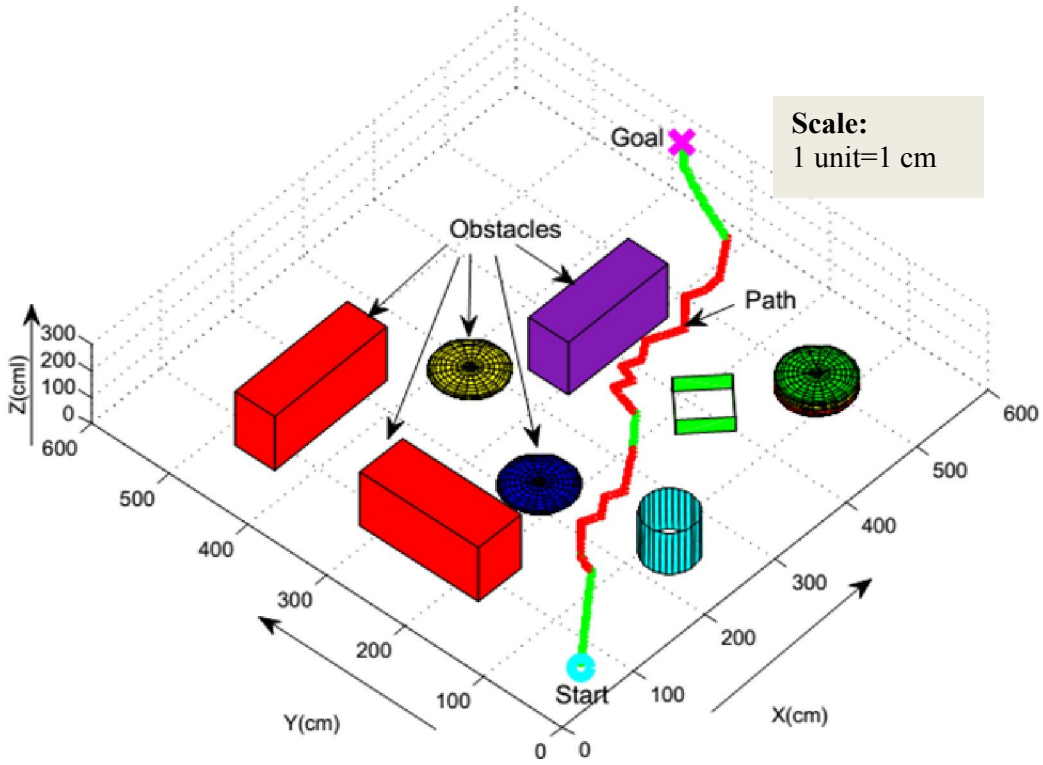


Figure 5.8: Simulation result with large number of obstacles

5.5.2 Selection of preferable population size for Adaptive SFLA:

Accuracy and stability of any stochastic metaheuristics based optimization can be severely influenced by population size of the algorithm [126]. Therefore, precise population size must be determined for modified version of SFLA which has been employed here to find out near optimal collision-free path for underwater robot in three-dimensional workspace of MATLB. Here, a quantitative analysis regarding the effect of change in population size of SFLA on navigational performance has been performed for simulated scenarios with three different obstacle arrangements. By varying population size, the variation in path length and obstacle avoidance behaviour for given three scenarios has been observed in Figures 5.9, 5.10 and 5.11. The observations of performed analysis have been documented in Tables 5.2, 5.3 and 5.4 respectively. In each scenario, population size has been varied from 30 to 300 and corresponding path length, obstacle avoidance and run time have been recorded for analysis purpose. The observations from performed qualitative analysis can be illustrated as follows:

- With the increase in population size, success in collision avoidance for underwater robot can be achieved with more perfection. But computation time required for

optimization will be significantly enhanced for rise in population size. Therefore, a proper trade-off between population size and run time for current navigational strategy must be found out.

- After a certain value, no change in collision avoidance behaviour has been observed for any increase in population size but run time linearly rises with the population size. Sometime large path length has also been found for high value of population size which is undesirable. It is preferable to find out an approximate value for population size which may be suitable to trace a near optimal path during three-dimensional motion.

In the three simulation scenarios (Figures 5.9, 5.10 and 5.11), it has been found that SFLA with population size of 120 has performed in a satisfactory manner regarding path length, obstacle avoidance and also computation time. Therefore, size of population for SFLA which contains next probable positions of underwater robot has been finalised as 120 and distribution of population has also been done by following structure of SFLA: 10 memplexes are assumed to be formed and each of which will contain 12 population members. Consecutively, 4 members are predicted to be grouped together to form a sub-memplex. So, 3 sub-memplexes will be present within each memplex.

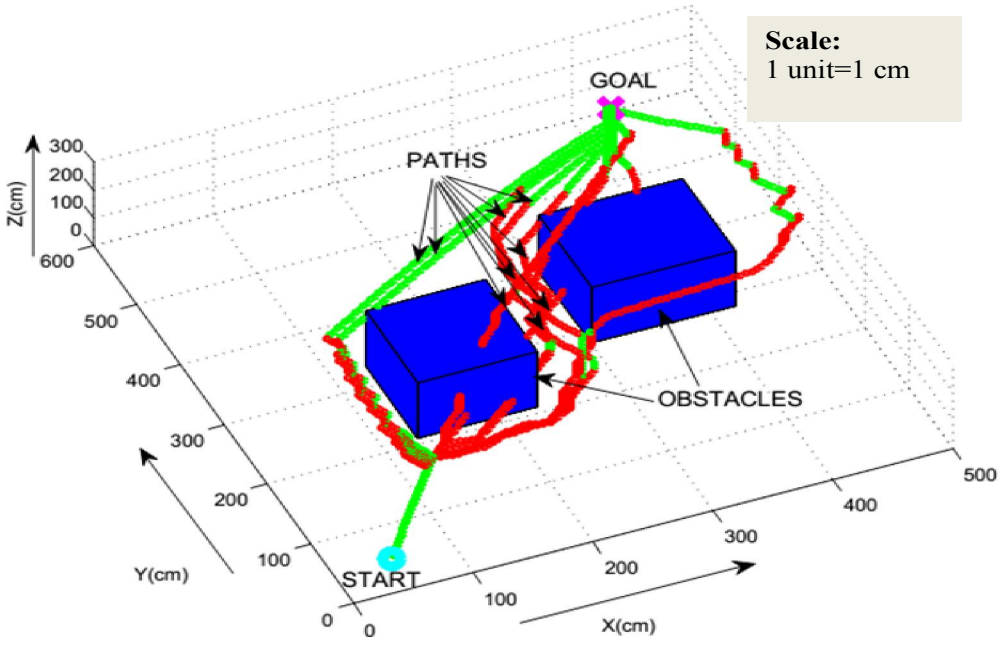


Figure 5.9: Simulation Result for Scenario1 with two obstacles

Table 5.2: Variation in Path Length and Run Time for Scenario1

Sl. No.	Population size for modified SFLA	Run time (in sec)	Path Length (in cm)	Collision avoided or not? (Yes/No)
1	30	9.34	675.9	No
2	60	9.89	694.7	No
3	90	10.85	682.2	No
4	**120**	11.63	750.4	**Yes**
5	150	12.79	749.7	Yes
6	180	13.47	749.2	Yes
7	210	14.19	752.3	Yes
8	240	14.88	751.7	Yes
9	270	15.17	749.6	Yes
10	300	16.03	752.9	Yes

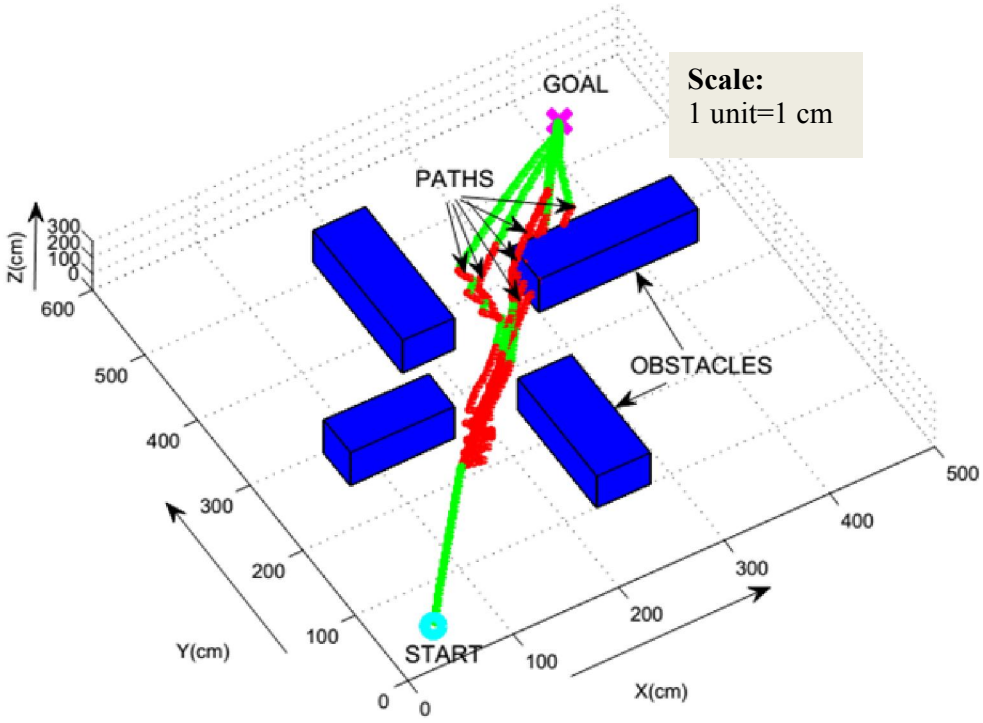


Figure 5.10: Simulation Result for Scenario 2 with four obstacles

Table 5.3: Variation in Path Length and Run Time for Scenario2

Sl. No.	Population size for modified SFLA	Run time (in sec)	Path Length (in cm)	Collision avoided or not? (Yes/No)
1	30	7.67	657.2	No
2	60	8.89	661.5	No
3	90	9.18	659	No
4	**120**	9.75	682.2	**Yes**
5	150	10.42	683.4	Yes
6	180	11.16	680.3	Yes
7	210	11.87	682.8	Yes
8	240	12.17	681.5	Yes
9	270	12.93	681.8	Yes
10	300	13.29	680.6	Yes

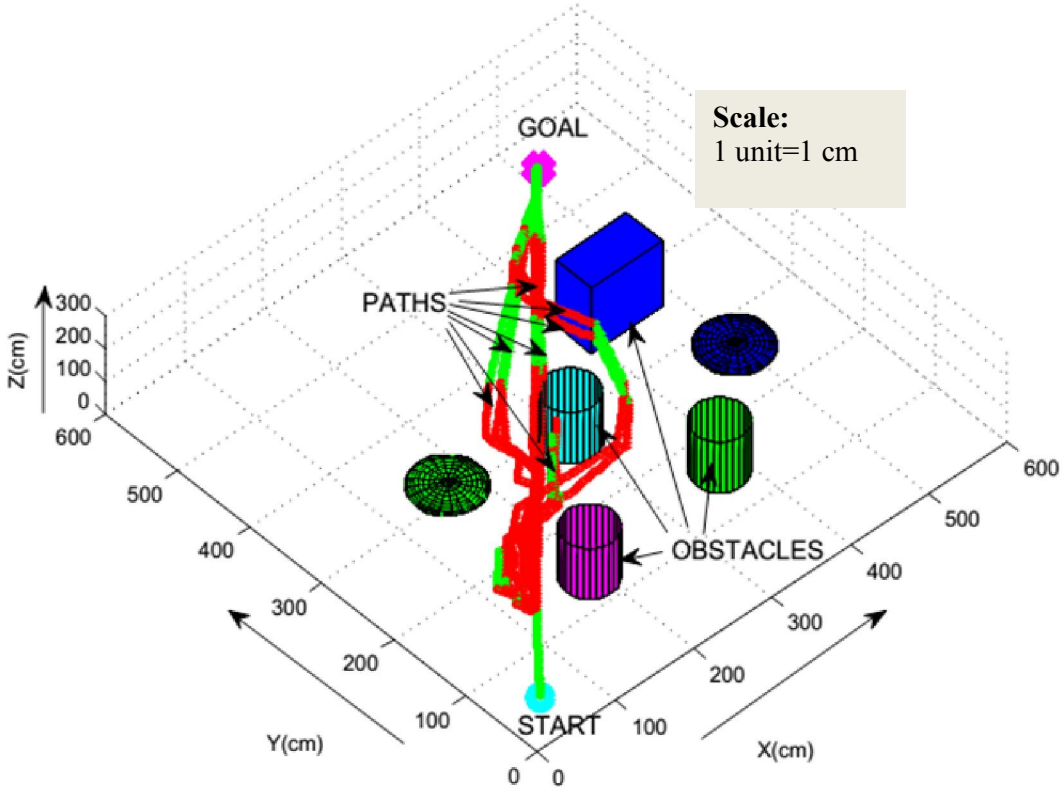


Figure 5.11: Simulation Result for Scenario 3 with six obstacles

Table 5.4: Variation in Path Length and Run Time for Scenario3

Sl. No.	Population size for modified SFLA	Run time (in sec)	Path Length (in cm)	Collision avoided or not? (Yes/No)
1	30	13.87	796.9	Indefinite Motion
2	60	14.95	791.2	No
3	90	15.18	802.1	No
4	**120**	16.25	826.3	**Yes**
5	150	17.32	827.6	Yes
6	180	17.96	829.5	Yes
7	210	18.37	828.2	Yes
8	240	19.12	828	Yes
9	270	19.93	829.4	Yes
10	300	20.38	828.7	Yes

5.5.3 Comparison with PSO, GA and SFLA for three-dimensional navigation:

A comparative simulation study has been computed here to authenticate optimization ability of proposed adaptive SFLA with respect to GA, PSO and conventional SFLA. Convergence speed, path length and travel time during three-dimensional navigation have been treated as performance indices for each optimization algorithm. In order to conduct a fair comparison in between considered metaheuristics, the major parameters of three-dimensional path optimization like fitness function (as defined by eq. 5.21), population size (P =120), decision variables for each solution vector (d=3), maximum iteration numbers (it_{max} = 200) etc. have been kept same as proposed adaptive SFLA based navigational approach (as given in Table 5.1).

Table 5.5: Details of control parameters for metaheuristics considered as underwater navigational strategies

Algorithms	Details of Control Parameters
PSO [168]:	Cognitive acceleration, c1=1, Social acceleration, c2 =2, Initial inertia weight ω_{min}= 0.9 and Final inertia weight ω_{max}= 0.1
GA [168]:	Crossover rate = 0.6, mutation rate = 0.2
SFLA [168]:	Evolution numbers in each memplex = 5 to 8
Proposed Adaptive SFLA:	Details of control parameters have already been mentioned in Table 5.1.

Table 5.5 contains specific values of control parameters for PSO, GA and conventional SFLA which have already been used by Aghababa [168] for three-dimensional path planning purpose. Adaptive SFLA, PSO, GA and conventional SFLA have been individually implemented as navigational strategy in given scenario of Figure 5.12 (a) for 20 independent runs. The most feasible path traced by each algorithm among its 20 runs has been portrayed in simulation result (Figure 5.12) and average variation in fitness value during navigation has also been derived for each algorithm as depicted in Figure 5.13. The observed convergence behaviour of algorithms has been discussed in following paragraph.

With the progress in iterations, average fitness values of all algorithms must be reduced to achieve the desired convergence. Adaptive SFLA has been found to be successfully converged on or before 50^{th} iteration with lowest fitness value of 137.73. Average convergence curve for PSO based navigational algorithm has also been reached stable fitness value of 143.36 near about 50^{th} iteration. But GA based path optimization algorithm requires more iterations than PSO or proposed Adaptive SFLA to converge at fitness value of 163.79 which is much higher than the lowest fitness value of PSO or Adaptive SFLA. Convergence speed of conventional SFLA has been recorded as little bit slower than the adaptive version of SFLA but lowest value of fitness for conventional SFLA (154.81) is more than that of the adaptive SFLA. From the current analysis, it can be deduced that Adaptive SFLA has better convergence speed than PSO though at the end of convergence, the optimal value of fitness has become almost same for both metaheuristics. Proposed Adaptive SFLA has shown dominance over GA regarding both convergence speed and fitness of optimal solution. Adaptive version of SFLA has also outperformed conventional SFLA with respect to run time and quality of solution.

Comparisons of considered metaheuristics based navigational approaches have also been carried out in Table 5.6 regarding path length and travel time for the simulation result of Figure 5.12. The optimum path length or straight distance between start and goal points of given scenario (Figure 5.12) has been measured as 614.25 cm. The deviation from optimal path length has been evaluated for each considered navigational strategy. Adaptive SFLA based navigational strategy has shown here superior performance than others. Safe avoidance from obstacles during navigation is mostly desirable for any path optimization method. Both PSO and Adaptive SFLA based navigational strategies have

shown better performance than GA and conventional SFLA in comparative study (Table 5.7) for simulation result of Figure 5.12.

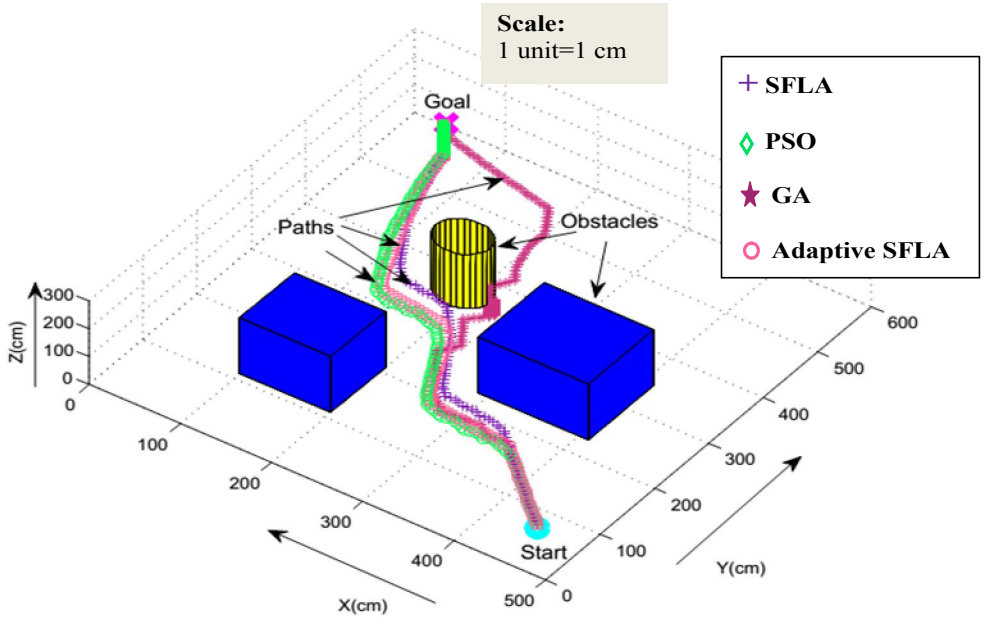


Figure 5.12: Simulated path of four different metaheuristics based navigational strategy

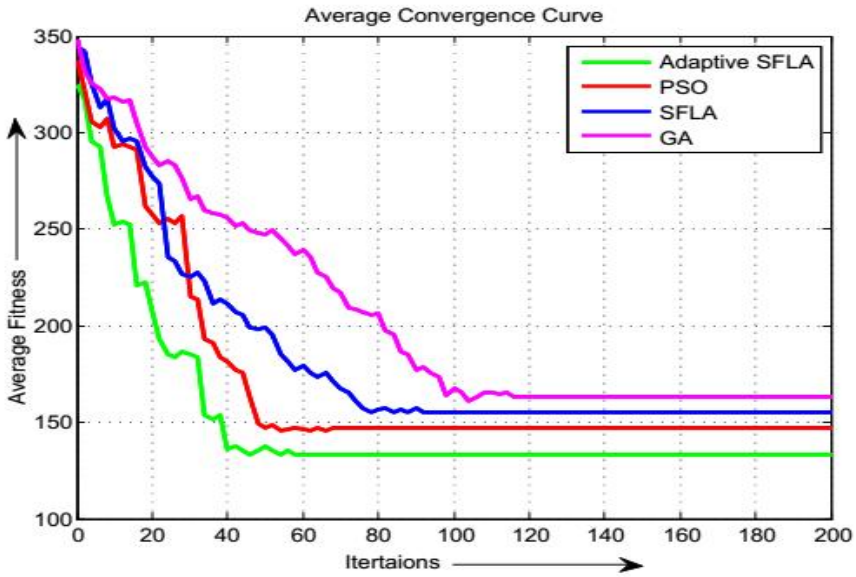


Figure 5.13: Average fitness convergence curves for navigational strategies while avoiding obstacles in given scenario of Figure 5.12

Table 5.6: Comparison between PSO, GA, SFLA and proposed Adaptive SFLA regarding path length and travel time for simulation scenario of Figure 5.12

Path Planning Algorithms	Path Length (in cm)	Deviation from Optimum path (in %)	Travel Time (in sec)	Safe from collision (Yes/No)
PSO [168]:	647.8	5.48	8.56	Yes
GA [168]:	691.2	12.52	11.77	No
SFLA[168]:	678.4	10.44	10.12	Yes
Proposed Adaptive SFLA:	644.3	4.91	8.51	Yes

5.5.4 Validation of proposed approach in comparison with other navigational strategies

In this section, navigational performance of the proposed Adaptive SFLA has been compared with other available methodologies of autonomous underwater navigation for validation purpose. Two comparative studies have been accomplished here regarding path optimization and collision avoidance ability of considered navigational strategies within three-dimensional simulation environment as follows:

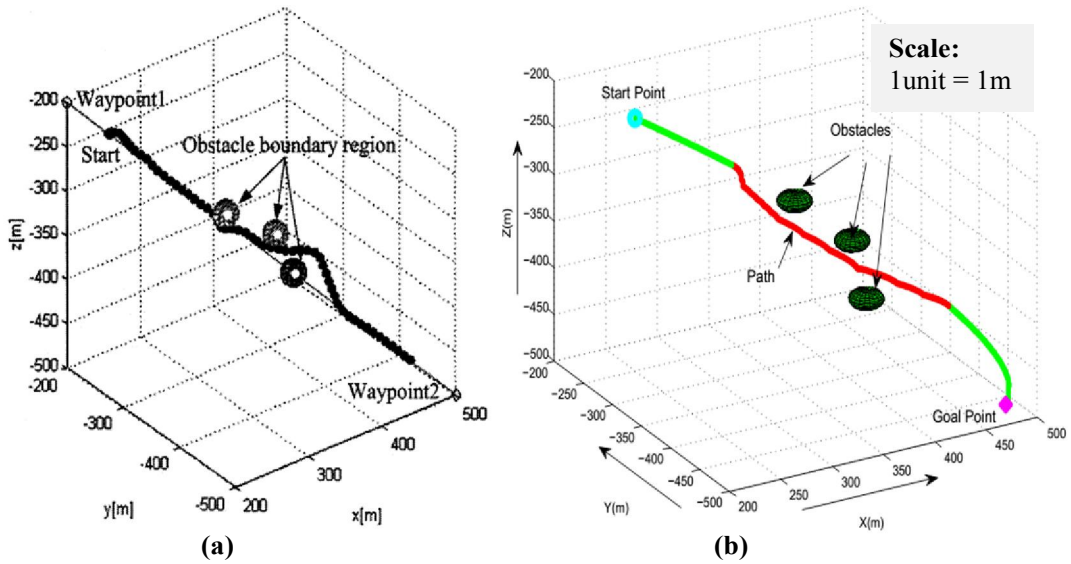


Figure 5.14: (a) Collision avoidance of MVFF algorithm as proposed by Kwon et al. [169] in presence of multiple obstacles; (b) Simulated path traced by proposed Adaptive SFLA based navigational approach for scenario same as Figure 5.14(a)

(a) Kwon et al. [169] have combined fuzzy logic approach with the Virtual Force Field (VFF) method which has been employed to track the predefined path for underwater robot and to achieve collision avoidance during autonomous navigation. The collision avoidance ability of Modified VFF method has already been exhibited in simulation environment (as shown in Figure 5.14 (a)) containing multiple static obstacles. Almost similar arrangement of obstacles, start and goal positions has been made in simulation environment of Figure 5.14(b) to assess the performance of recently proposed Adaptive SFLA based navigational strategy in that particular environment. From the visual comparison between simulation results of Figure 5.14(a) and (b), it can be found that more clearance from obstacles has been achieved by the proposed Adaptive SFLA than the navigational scheme based on modified VFF method [169]. The comparison between two mentioned navigational strategies regarding path length which has been executed in Table 5.7 shows the supremacy of proposed Adaptive SFLA based path planning over Modified VFF method [169] for the given scenario.

(b) Liu et al. [170] have introduced three-dimensional path planning algorithm based on modified version of firefly algorithm. The simulation results have also been shown to test whether the convergence speed of proposed algorithm is suitable for three-dimensional path planning or not. To represent real marine environment in simulation mode, Liu et al. [170] have fixed the starting point at (111.30^0 E, 15.87^0 N) with the depth of 230 meters and the destination point at (112.80^0 E, 17.11^0 N) with the depth of 450 meters. Between these two points, multiple obstacles have been distributed in a scattered manner at different depths and point like underwater robot has been used to trace path from start to goal as shown in Figure 5.15(a). For comparison purpose, an attempt has been made to create a simulation scenario in Figure 5.15(b) where obstacles and start and goal points of underwater robot are at near about same positions like Figure 5.15(a). To execute an impartial comparison, some assumptions has been made here to represent the three axes of developed simulated environment as follows: for X-Y axis: 1unt = 0.001^0 and for Z axis: 1unit = 200 m of depth. Comparison between navigational performances of recently proposed Adaptive SFLA and modified firefly algorithm in given simulated scenario (Figure 5.15) has shown good agreement regarding their obstacle avoidance abilities.

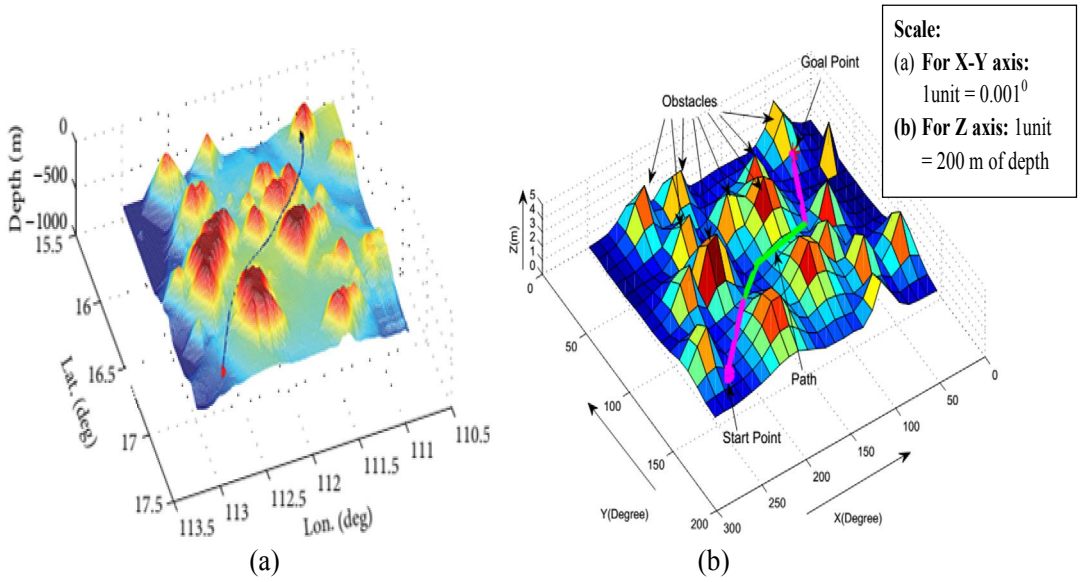


Figure 5.15: (a) Simulated path traced by AMFA algorithm as proposed by Liu et al. [170] in presence of multiple obstacles; (b) Simulated path traced by proposed Adaptive SFLA based navigational approach for scenario same as Figure 5.15(a)

The length of simulated path traced by respective path planning algorithm has been considered as performance index in this comparative study. Table 5.7 shows that adaptive SFLA based navigational method has traced shorter path than AMFA [170]. Such comparative study can ensure the robustness and competence of proposed Adaptive SFLA as three-dimensional path planning algorithm.

Table 5.7: Numerical Comparison between Proposed Adaptive SFLA algorithm and other navigational strategies for three-dimensional simulated environments

Figure No.	Navigational Strategy	Length of 3D path traced by underwater robot (in cm)	Deviation (in %)
5.14(a)	Modified VFF method [169]	1053.5	2.4
5.14(b)	Proposed Adaptive SFLA	1033.6	
5.15(a)	Modified Firefly algorithm [170]	541.5	1.73
5.15(b)	Proposed Adaptive SFLA	535.7	

5.6 Experimental Verification of Simulation Result

The navigational performance of proposed path planning algorithm during simulation study has been justified by performing real-time experiment of underwater robot GNOM-baby in swimming pool environment. Features of "GNOM baby" has been incorporated in Appendix-A. In experimental mode (Figure 5.16), start and goal points and arrangement of obstacles with different shapes and sizes have been done within swimming pool area by following the simulation scenario of Figure 5.8. It has been observed that underwater robot GNOM-baby embedded with proposed Adaptive SFLA based navigational strategy has successfully traced a collision-free near optimal path from start to goal during experiment. Figure 5.16 shows different intermediate stages of navigation by underwater robot GNOM-baby within real time swimming pool environment. Stage 1 depicts the view of experimental environment which contains static obstacles at different positions, goal point and underwater robot rested at start point. From stage 2 onwards, underwater robot GNOM- Baby has started to move towards goal. In successive stages (stage 3 to 5), underwater robot has successfully avoided nearest obstacles and also progresses forward. At stage 6, motion of underwater robot has been stopped as goal position has already been reached.

5. 6.1 Comparison between simulation and experimental results:

Metaheuristics based path planner has to be applied for multiple times on same scenario to ensure whether the path followed by robot is optimal or not. Due to stochastic nature, the solution vectors of two consecutive iterations which are randomly generated within population of proposed Adaptive SFLA by following eq. 5.2 may not be same by both value and direction. So, global best positions chosen by adaptive SFLA based navigational strategy in each run of algorithm may also be different from each other. In a consequence, both path length and time taken by robot can be varied in successive runs of proposed path planner. Therefore, simulation study (Figure 5.8) and its corresponding experimental verification (Figure 5.16) has been repeatedly performed for 20 times within same scenario using proposed Adaptive SFLA. Path length and travel time for each run of path planning algorithm in simulated and experimental modes have been noted in Table 5.8 and 5.9 respectively. Without considering any obstacle, straight distance between start and goal point of given scenario has been considered as optimal path length which has been recorded as 664.7 cm and 707.3 cm for simulation (Figure 5.8) and experimental (Figure 5.16) results respectively.

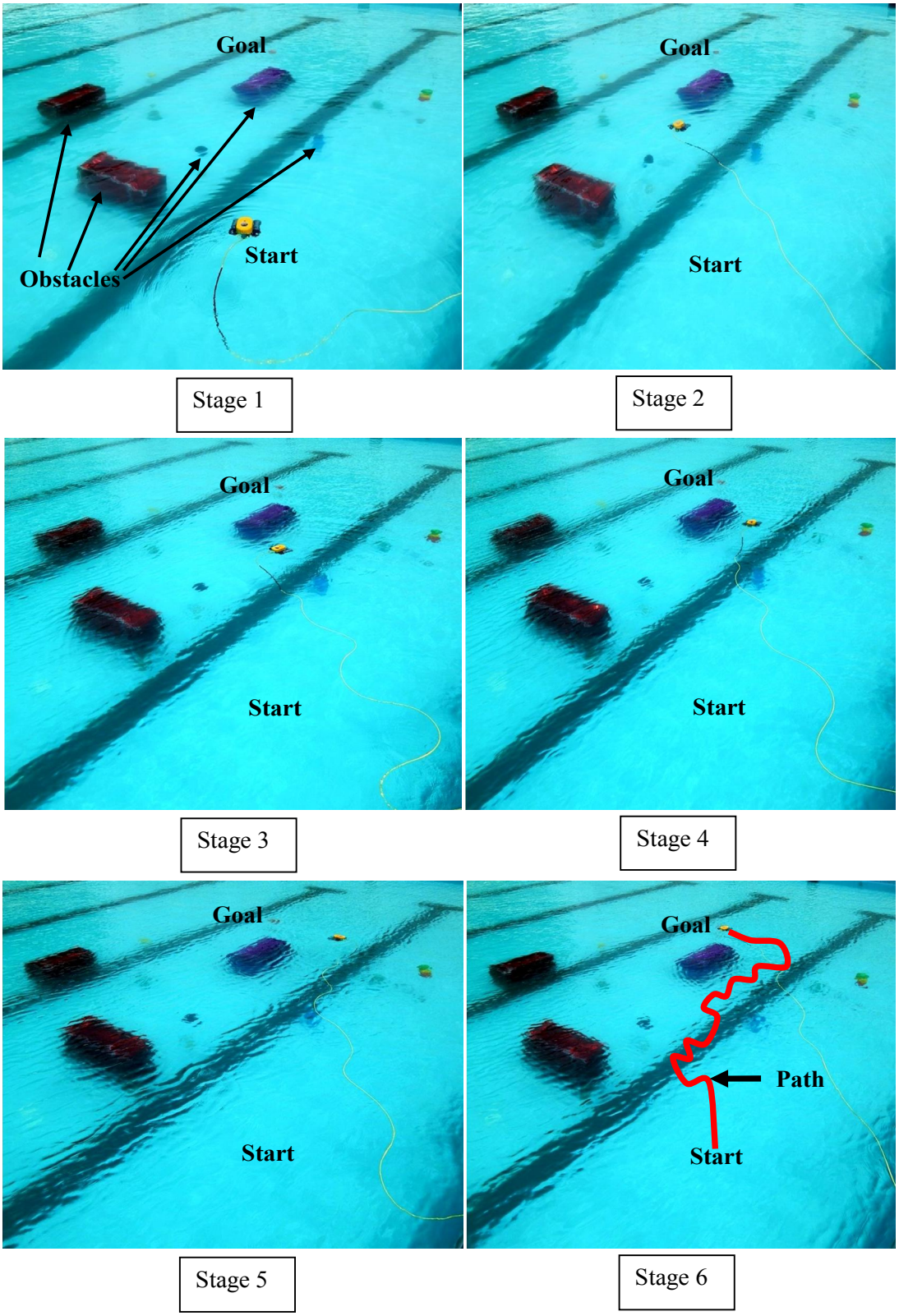


Figure 5.16: Experimental view for GNOM Baby embedded with Adaptive SFLA based Navigational Strategy for obstacle arrangement near about same as in Figure 5.8

To avoid obstacles, actual path length traced by underwater robot must be higher than the optimum one. But proposed navigational strategy has attempted to minimize the path length near about its optimum value. Deviations in the length of generated paths with respect to optimum path have been observed in Table 5.8. The percentage error between simulated and experimental path length has also been computed for each run. For both simulated and real time environments, paths with minimum length are considered as best paths which have been shown in Figure 5.8 and 5.16 respectively. Shortest path has been obtained in simulation mode at 10^{th} run and in experimental mode at 12^{th} run as shown in Table 5.8. Average errors between simulated and experimental results regarding path length and travel time have been calculated in Table 5.8 and 5.9 respectively.

Table 5.8 Comparison between experimental and simulated results for path length in Scenario1 (Figure 5.8 and 5.16)

1	2	3	4	5	6
No. of Runs	**Path length in simulated mode (in cm)**	**Deviation from Optimum path (%)**	**Path length in experimental mode (in cm)**	**Deviation from Optimum path (%)**	**Difference between column 2 and 4 (%)**
1^{st}	706.7	6.32	752.1	6.32	5.94
2^{nd}	712.2	7.14	749.8	6.01	5.72
3^{rd}	705.8	6.19	745.7	5.42	5.62
4^{th}	710.7	6.91	750.7	6.12	5.87
5^{th}	697.2	4.88	739.5	4.51	5.92
6^{th}	711.6	7.05	751.6	6.25	5.69
7^{th}	708.8	6.64	749.8	5.91	5.78
8^{th}	704.5	5.99	746.3	5.46	5.8
9^{th}	710.3	6.86	752.2	6.35	5.67
10^{th}	**693.7**	**4.36**	737.1	4.20	5.85
11^{th}	702.2	5.64	737.4	4.25	4.94
12^{th}	696.9	4.84	**735.9**	**4.03**	5.72
13^{th}	703.5	5.84	741.5	4.83	5.45
14^{th}	699.9	5.29	738.1	4.35	5.78
15^{th}	701.1	5.48	736.1	4.07	4.31
16^{th}	697.2	4.89	738	4.33	5.03
17^{th}	705.7	6.17	737.4	4.25	5.87
18^{th}	708.3	6.55	748.3	5.79	5.51
19^{th}	698.8	5.12	739.7	4.57	5.92
20^{th}	695.9	4.69	736.9	4.18	5.53
Average difference in path length between simulated and experimental results					5.63

Table 5.9 Comparison between experimental and simulated results for travel time in Scenario1 (Figure 5.8 and 5.16)

1	2	3	4
No. of Runs	**Travel time in simulated mode (in sec)**	**Travel time in experimental mode (in sec)**	**Difference between column 2 and 4 (in %)**
1st	11.22	11.75	4.76
2nd	15.48	16.41	5.98
3rd	10.86	11.42	5.16
4th	13.94	14.81	6.25
5th	8.83	9.29	5.23
6th	14.52	15.27	5.13
7th	12.89	13.65	5.88
8th	9.65	10.23	6.03
9th	13.66	14.41	5.50
10th	7.80	8.25	5.89
11th	9.12	9.66	5.87
12th	7.83	8.30	5.95
13th	9.26	9.72	4.98
14th	8.05	8.46	5.09
15th	8.88	9.30	4.82
16th	7.92	8.37	5.69
17th	10.53	11.15	5.87
18th	12.43	13.10	5.41
19th	8.22	8.69	5.69
20th	7.91	8.36	5.73
Average difference in travel time between simulated and experimental results			5.55

Table 5.10 Comparisons between experimental and simulated results for five more scenarios regarding path length

Different Scenarios	Best path length in Simulation (in cm)	Best path length during Experiment (in cm)	Difference (in %)
Scenario-2	639.9	671.2	4.86
Scenario-3	627.2	663.4	5.77
Scenario-4	587.4	614.5	4.6
Scenario-5	726.5	765.3	5.34
Scenario-6	615.2	647.1	5.17

To certify the path optimization ability of the proposed method irrespective of environmental condition, a number of simulations and their corresponding experimental verifications have been performed by employing proposed Adaptive SFLA based navigational strategy for different obstacle arrangements. Comparisons of simulation and experimental results have been executed for five different scenarios in terms of path length and travel time and the results are depicted in Table 5.10 and 5.11 respectively. Minor differences have been recorded in all comparisons between simulation and experimental results. Therefore, it can be stated that proposed navigation strategy has been authenticated up to a satisfactory level.

Table 5.11 Comparisons between experimental and simulated results for five more scenarios regarding travel time

Different Scenarios	Travel time during simulation (in sec)	Travel time during experiment (in sec)	Difference (in %)
Scenario-2	7.03	7.41	5.37
Scenario-3	6.89	7.26	5.31
Scenario-4	6.18	6.54	5.83
Scenario-5	8.16	8.60	5.40
Scenario-6	6.76	7.15	5.75

5.7 Summary

A new adaptive version of memetic evolution based optimization method has been proposed and implemented in the current chapter to achieve the near optimal safe path during underwater navigation. On detection of obstacle by on-board sensors, underwater robot has been stimulated to follow the next global best positions as chosen by proposed population based Adaptive SFLA in a sequential manner. Fitness function for proposed methodology has been designed by integrating different criteria like obstacle avoidance and path length minimization. The iterations for selecting best positions continue until the robot reaches the goal point of given scenario. On the basis of performed simulation and experimental investigations, some precious aspects of proposed Adaptive SFLA based three-dimensional underwater navigation has been briefed here:

- A new acceleration term or scaling factor which is adaptive by nature with respect to the change in fitness of population and iteration numbers has been introduced in frog leaping rule to enhance the fine tuning ability of the memetic evolution based local

search. Direction of leaping or motion has also been tried to be controlled by adding adaptive deviation angle in leaping step size during local search.

- After shuffling of memplexes, slight adaptation in global best solution has been found to be advantageous for maintaining the diversity in population even at last stage of iterations. So, the probability of premature convergence can be successfully diminished.

- Proposed adaptive SFLA based navigational strategy has exhibited reasonably effective performance while avoiding obstacles, escaping from traps and seeking goal within complex three-dimensional simulated scenarios of MATLAB.

- A number of simulated experiments have been performed on a trial and error basis to finalize a cost effective population size for proposed memetic metaheuristics.

- Adaptive SFLA based navigational strategy has shown the faster convergence speed than popular metaheuristics like PSO, GA and conventional SFLA while tracing path in three-dimensional simulated environment. Comparative study regarding path length, travel time and collision avoidance ability has ascertained the ascendancy of proposed adaptive version of memetic metaheuristics over mentioned evolutionary algorithms during three-dimensional navigation within given simulation setup.

- Proposed adaptive SFLA based navigational strategy has been compared with modified versions of conventional VFF method [169] and firefly algorithm [170] based three-dimensional navigational strategies in terms of path optimization ability. Comparison has shown a good agreement.

- For each simulation and corresponding experimental scenario, a number of trials have been carried out by employing proposed Adaptive SFLA as three-dimensional navigational strategy. Performance of proposed three-dimensional path planner in each trial has been evaluated based on time taken and path length covered by underwater robot while moving from start to goal within given scenario. In average, the error between simulated and experimental results has been found to be within the range of 5-6% (5.23% for path length and 5.53% for travel time) which can be considered as reasonable for practical application.

Therefore, proposed Adaptive SFLA can be successfully implemented as a three-dimensional path planner in an unpredictable underwater environment. In subsequent chapters, different stochastic metaheuristics based navigational strategies for underwater robot have been proposed and implemented to achieve robotic behaviours in simulated and experimental scenarios.

6. Dynamic Differential Evolution based Navigational Strategy for Underwater Robot

Path planning for underwater robot based on memetic evolution has already shown satisfactory performance in previous chapter. Another unique concept of metaheuristics, where evolution strategy performs simple arithmetic operations on population members, has been addressed in this chapter to solve path optimization problem of underwater robot in a realistic manner. Reduction in computational complexity and increase in degree of optimality in three-dimensional navigational algorithm have been considered as major objectives for this new attempt of present research work.

6.1 Introduction

To execute precarious marine tasks, underwater robot needs autonomous navigation to manoeuvre from one point to another within workspace without any human intervention. Choice of waypoints must aid the minimization of path length travelled by underwater robot [5]. In this chapter, differential evolution (DE) algorithm has been employed as an optimizer to find out robot's global best pose during navigation. Storn and Price [120] have introduced DE as simplified and modified version of GA which is more deterministic and less stochastic by nature than other EAs. DE requires few or no assumptions about the problem to be optimized. Advantages of DE such as simplicity, effortless execution, wide/fine exploitation ability and faster convergence speed have found to be effective while solving various nonlinear optimization problems. In large scale application, DE can provide desirable faster convergence speed. But metaheuristics like DE may suffer from premature convergence, lower degree of optimality and uncertainty in search direction. Therefore, self-learning ability has been incorporated in new version of DE for solving path optimization problem of underwater robot. Simulation and experimental investigations have been executed to evaluate navigational performance of proposed dynamic version of DE based optimization method.

6.2 Basic Features of Differential Evolution Algorithm

Among EAs, Differential Evolution algorithm has some unique features of simple arithmetic operators which take part in classical processes like mutation, crossover and selection, (Figure 6.1) to reduce computational complexity. DE generally starts with a

randomly initialized population set of solution vectors $\{v_p;p=1,2,..P\}$ with d dimensions. Fitness function for vectors has to be defined as based on optimization objectives of the given problem. As first step of iteration, one population member termed as target vector is randomly selected to be evolved. In mutation operation, another vector from population termed as donor and difference of one or more pairs of vectors are chosen randomly. Difference vector for each dimension is adjusted by multiplying amplification factor F_i and then added to donor for evolving new mutated vector [120] (as shown in Figure 6.1). This easiest mechanism of mutation ensures automatic scaling of newly generated vectors towards convergence [171].

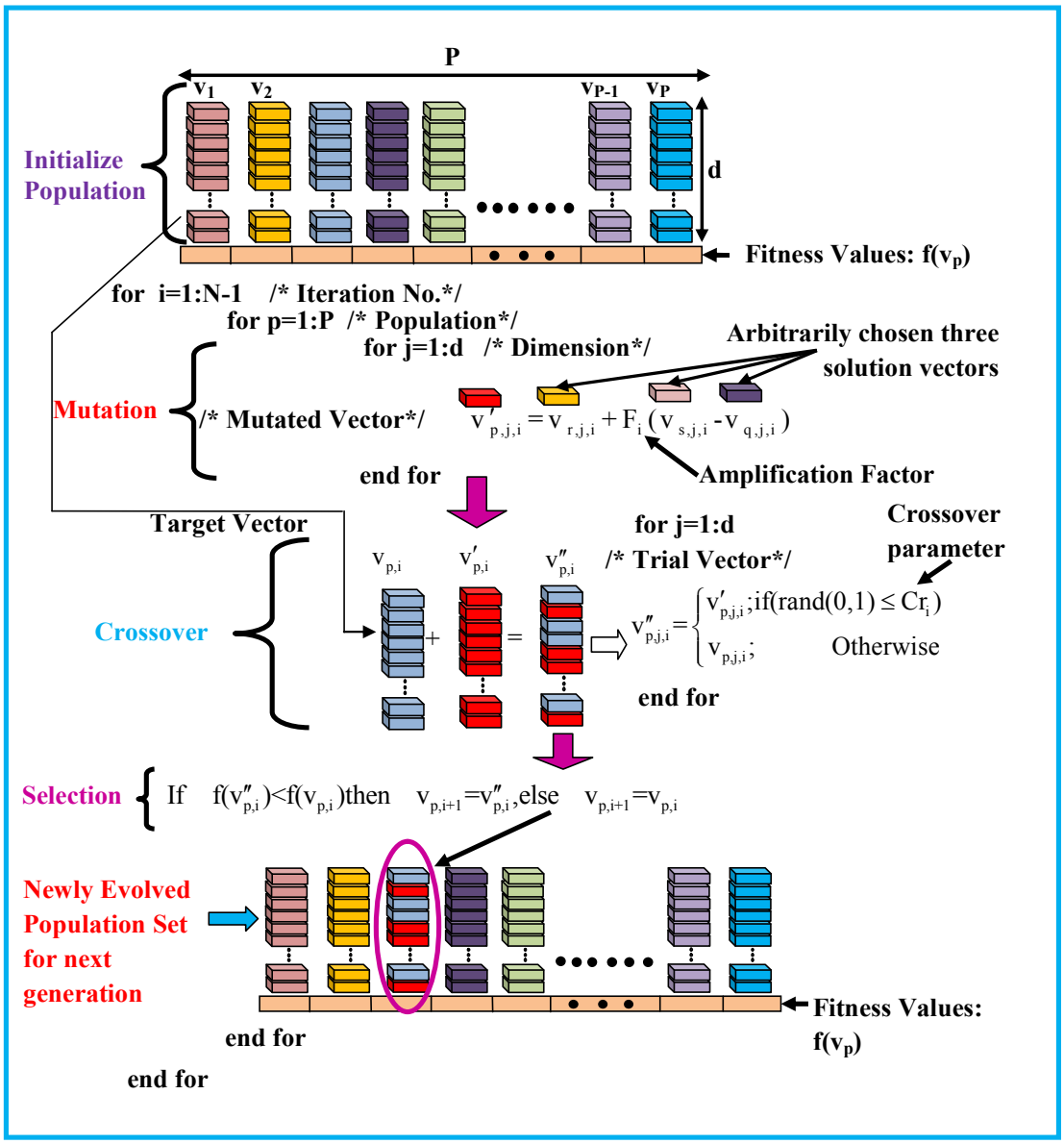


Figure 6.1 Description of Basic Differential Evolution Algorithm

The crossover operation recombines the parameters of each mutant vector with those of the target vector with probability of CR_i (crossover constant) to reproduce a child or trail vector (given in Figure 6.1) which must lie within specified boundaries. Opposite to mutation, recombination provides diversity in population by adding newly generated parameters in vectors [172]. The selection process of survivor for next iteration follows the Darwinian principle of "survival of the fittest". Vector with less fitness value, which may be trial or targeted one, will be included in population to be used in next iteration. DE normally improves the fitness value of population vectors, in worst cases, keeps them constant but never allows degradation [123]. Population size and dimensions of solution vectors has to be maintained as constant throughout the iterations of DE.

6.3 Importance of control parameters and mutation strategies in DE algorithm

The values of control parameters decide how quickly the algorithm will converge. A balance between local and global search abilities of DE based optimization can be achieved by varying the control parameters (F and CR) throughout the iterations in a regulated way. Based on dimension of given optimization problem, accuracy in selection of population size P may also contribute in increase of diversity of the search space. Fitness function and termination criteria must be previously defined and will remain constant for all generations of solution vectors.

Storn and Price [120], the originator DE algorithm, have advised some specific values of these three control parameters to be constant in iterations for satisfactory performance of DE. As per suggestions, population size, $P \in (5d,10d)$, where d is dimension of population member; initially, F=0.5 is a proper choice and as iterations progress F may vary between (0.4, 1.0); CR can be fixed at 0.1 or 0.9. Gämperle et al. [173] have solved different benchmark functions, with: $3d \le P \le 8d$, F=0.6 and $0.3 \le CR \le 0.9$. Ronkkonen et al. [174] have used: $P \in (2d, 4d)$, $F \in (0.4, 0.95)$ and $CR \in (0.0,1.0)$. Kaelo and Ali [175] have employed F randomly distributed in [-1, -0.4] U [0.4, 1]. Karaboga and Akay [176] have suggested $3d \le P \le 10d$ and F=0.6 to get optimum solution in engineering problems.

So, it can be stated that F within range of (0, 2) has been preferred by researchers. Adaptation in F allows stochastic variation in the amplification of difference vector. Low value of F reduces the amplitude of the difference vector resulting less number of mutated

vectors outside the range but it may adversely affect diversity of global optimization. In reverse way, increment in difference vector may generate trial vectors beyond the prescribed range but convergence speed may increase. Crossover probability controls mutated and target vector in recombination operation. Zaharie [177] has explored the effect of recombination and its probability factors on success rate of mutation strategy in DE. For CR=1.0, all dimensions of child vector will be same as newly mutated vector resulting increase in diversity of search space. As CR decreases, the probability of parameters of parent vector to be included in child vector increases.

The donor or base vector within mutation strategies of DE may be random vector from population or target vector itself or best vector of current population. The difference vector which will be scaled by amplification factor can be computed from one or more pairs of different population members. For regular DE operation, at least five types of mutation strategies are available to evolve solution vectors as defined by [178]:

"DE/rand/1": $v'_{p,i} = v_{r,i} + F \cdot (v_{s,i} - v_{q,i})$ (6.1)

"DE/best/1": $v'_{p,i} = v_{best,i} + F \cdot (v_{r,i} - v_{s,i})$ (6.2)

"DE/rand-to-best/1": $v'_{p,i} = v_{p,i} + F \cdot (v_{best,i} - v_{p,i}) + F \cdot (v_{r,i} - v_{s,i})$ (6.3)

"DE/rand/2": $v'_{p,i} = v_{r,i} + F \cdot (v_{s,i} - v_{q,i}) + F \cdot (v_{u,i} - v_{t,i})$ (6.4)

"DE/best/2": $v'_{p,i} = v_{best,i} + F \cdot (v_{r,i} - v_{s,i}) + F \cdot (v_{u,i} - v_{t,i})$ (6.5)

Where, $v'_{p,i}$: Mutated vector; $v_{r,i}, v_{s,i}, v_{q,i}, v_{u,i}, v_{w,i}$ are the randomly chosen vectors from population and must be different from vector chosen as target $v_{p,i}$; F magnifies the amplitude of difference vectors; $v_{best,i}$: vector in population with best fitness value for current iteration.

Each strategy can be associated with any one of two crossover mechanisms (binomial and exponential). Hence, total ten mutation strategies are available in DE algorithm [179]. Some of them may focus on exploration and others may provide better convergence. Simple arithmetic operation based evolution strategy of DE has received from significant interests from diverse research areas. DE based search mechanism has shown faster

convergence speed along with more certainty than other global optimization method [180]. Least number of control variables and parallel search ability of DE have found to be robust enough for providing solution towards real world problems. Analysing different applications of DE, it can be specified that the selections of mutation strategy and control parameters are highly dependent on the problems to be optimized [181]. Time-consuming trial and error approach for tuning of control parameters as well as selection of mutation scheme has been attempted to avoid by employing numerous versions of DE throughout past decades. While generating new solution in DE, limited consideration of population may undesirably affect the diversity of search process resulting in local minima situation. To balance global and local search abilities, a new version of DE has been proposed and implemented as a three-dimensional navigational strategy.

6.4 Framework of proposed Dynamic Differential Evolution

Improper selection of control parameters and mutation strategies for DE algorithm may lead the search towards an uncertain direction resulting in huge computational complexity. Search of suitable mutation strategy and control parameters of DE at different stages of optimization is quite tiresome [182]. The optimization performance and convergence speed of DE has been proposed to be improved by inducing some dynamic nature in evolution steps. Apart from dynamic adaptation of control parameters, selection of mutation schemes has also been varied at different stages of search process in an autonomous manner. Sequential steps of proposed Dynamic DE algorithm has been illustrated as follows:

Initialization

Initialization of control parameters such as Population size: P, No. of decision variables: d, Maximum and Minimum values of F and CR: $\{F_{min}, F_{max}\}$ and $\{CR_{min}, CR_{max}\}$ and Initial values for control parameters: F_1 and CR_1

Population of DDE algorithm $\{S_p:v_{j,p}\}$ can be initialized as follows:

For p=1: P

For j=1: d

$$v_{j,p} = v_{j,p}^{\min} + rand(0,1)\bullet(v_{j,p}^{\max} - v_{j,p}^{\min}) \tag{6.6}$$

End for

End for

Where, j=1,2….,d; dimension of each population member, p=1,2,…..P; number of population members and rand(0,1): randomly generated number in between 0 and 1; $v_{j,p}^{max}$ and $v_{j,p}^{min}$ are upper and lower limits for decision variable of solution vector respectively.

Fitness Evaluation

Defined objective function of specified optimization problem has been employed to evaluate fitness of each population member as follows:

For p=1: P

Find out $f(v_p)$ =fitness of v_p (6.7)

End for

After completion of initialization and fitness evaluation process, iterations for optimization will be started. In each iteration, all population vectors will be evolved once through three steps: mutation, crossover and selection. Pseudo code for these three steps of Dynamic DE has been elaborated in Figure 6.2.

Dynamically Adaptive Mutation Process

A dynamic mutation rule has been proposed here to improve local exploration ability and convergence rate of DE based optimization. It is combination of three mutation strategies: DE/worst to best /1, DE/target to best/2 and DE/rand/2 as shown in Figure 6.2. Selection of mutation scheme has been done based on the success rate of mutated vectors (MSR) in crossover process (eq. 6.14). In early iterations, as MSR may be low, the probability of using basic scheme (DE/rand/2) (eq. 6.10) will be high favouring exploration or global search ability of algorithm. But it gives slow convergence. As iteration progresses, MSR will be high and the choice has to be made between DE/ worst to best /1 and DE/target to best/2 based on the current iteration number. For $i \ll i_{max}$, chances of selecting DE/target to best/2 (eq. 6.9) will be high. As the considered mutation strategy is a combination of random difference and difference between best fitted vector and target vector, it may balance both exploration and exploitation abilities of search process. With the increase of iterations $i \leq i_{max}$, probability of both strategies (eq. 6.8) and 6.9) will be approximately same. Therefore, the direction of search may get deviated from exploration to exploitation.

/* Iterations for optimization based on Dynamic Differential Evolution */

for $i = 1 : i_{\max}$

for $p = 1 : P$

for $j = 1 : d$ /*Mutation*/

if $MSR > rand(0\ ,1)$

if $u(0,1) \geq \frac{i_{\max} - i}{i_{\max}}; then \quad v'_{j,p,i} = v_{j,p,i} + F_i \cdot (v_{j,best,i} - v_{j,worst,i})$ (6.8)

else $v'_{j,p,i} = v_{j,p,i} + F_i \cdot (v_{j,best,i} - v_{j,p,i}) + F_i \cdot (v_{j,r,i} - v_{j,s,i})$ (6.9)

endif

else $v'_{j,p,i} = v_{j,r,i} + F_i \cdot (v_{j,s,i} - v_{j,q,i}) + F_i \cdot (v_{j,u,i} - v_{j,t,i});$ (6.10)

endif

endfor

Set $MSR = 0$

for $j = 1 : d$ /*Crossover*/

if $(rand(0,1) \leq Cr_i); then \quad v''_{p,j,i} = v'_{p,j,i};$ (6.11)

$msr++;$ (6.12)

else $v''_{p,j,i} = v_{p,j,i};$ (6.13)

endif

endfor

$MSR = \frac{msr++}{3};$ (6.14)

If $fitness(v''_{p,i}) < fitness(v_{p,i})$ /*Selection*/

then $v_{p,i+1} = v''_{p,i};$ (6.15)

$sr++;$ (6.16)

else $v_{p,i+1} = v_{p,i};$ (6.17)

endif

endfor

$accep\tan ce \quad rate = \frac{sr++}{P};$ (6.18)

endfor

Figure 6.2: Adaptive Mutation, Crossover and Selection Mechanisms for DDE

As iteration increases more, $i \approx i_{\max}$, DE/worst to best/1 (eq. 6.8) will be mostly chosen for providing faster convergence by directing all vectors towards best fitness solution. As per given Pseudo code of Figure 6.2, only one mutation scheme will be actuated for one vector in one generation. Such dynamic adaptation of mutation schemes may reduce the computational time for proposed DDE.

Crossover

Based on a comparison between a random number and crossover probability (CR_i), a trial or child vector has been regenerated by combining parameters of mutated as well as target vectors in eq. 6.11 or 6.13. Only binomial type crossover mechanism has implemented here. On selection of parameter from mutated vector, the success rate of mutation process (msr) will be increased in eq. 6.12.

Selection

Target vector of population may be replaced by newly generated offspring when fitness of offspring vector is better than target vector as given in eq. 6.15. The selection rate (sr) will also be enhanced by following eq. 6.16. In other case, target vector will remain unchanged in population set as per eq. 6.17. At the end of iteration, overall rate for acceptance of newly evolved solution vector has been recorded by using eq. 6.18.

Adaptation of F and CR

As such there is no systematic way for determining the value of control parameters (F_i CR_i) and their optimal values are also typically problem-specific. A unique adaptation mechanism for control parameters has been proposed here and described as follows:

If acceptance rate > 0.2

$$F_{i+1} = F_{\min} + \frac{f_{i,\max}}{f_{i,\min}} \bullet \frac{i}{i_{\max}} \bullet (F_{\max} - F_{\min}) \tag{6.19}$$

$$CR_{i+1} = CR_{\min} + \frac{1}{accep\tan ce \quad rate} * \frac{i}{i_{\max}} * (CR_{\max} - CR_{\min}) \tag{6.20}$$

Else,

$$F_{i+1} = F_{\max} \text{ and } CR_{i+1} = CR_{\min} \tag{6.21}$$

End if.

At early stage of iterations, acceptance rate has been expected to be lower than its threshold value. So, F_{max} and CR_{min} will be employed for next iteration by following eq. 6.21 to provide wide search ability in optimization process. During initial iterations, large value of F_i associated with DE/rand/2 mutation strategy may enhance the diversity in

population and inversely, small value of CR_i may limit the diversity within boundary. As success rate increases, value of F_i will be decided based on fitness differences between two consecutive iterations (eq. 6.19) and CR_i will also be regulated by iteration number and acceptance rate by following eq. 6.20. With the progress of iterations, fine tuning of F_i and CR_i may accelerate the two mutation schemes (eq. 6.8 and eq. 6.9) to provide faster convergence along with better quality of solution.

Boundary constraint of each newly generated parameter of mutated vectors has to be verified. The parameter outside the boundary will be reset by following the rule as given below:

$$v'_{j,p,i} = \begin{Bmatrix} v_{j,p}^{\min} + rand(0,1)(v_{j,p}^{\min} - v'_{j,p,i}); if \quad v'_{j,p,i} < v_{j,p}^{\min} \\ v_{j,p,i}^{\max} - rand(0,1)(v'_{j,p,i} - v_{j,p}^{\max}); if \quad v'_{j,p,i} > v_{j,p}^{\max} \end{Bmatrix} \tag{6.22}$$

Iterations may be continued up to $i=i_{max}$ or may get stopped when the fitness differences between best fitted population vectors of consecutive iterations are almost negligible.

6.5 Implementation of DDE as path planning algorithm

During underwater robot's motion towards desired location, online information from on-board sensors can be conveyed to the controller for a certain range of distance and angle. On detection of obstacles within threshold range of sensors for current position of robot, proposed Dynamic DE based algorithm has been employed to choose robot's next best pose within limited search space (as given in flowchart of Figure 6.3). While executing initial step of DDE, the population of next possible poses for underwater robot has been randomly generated by following the mechanism as described in eq. 5.15 (Figure 5.5) of Section 5.4.

Table 6.1: Parameters for Dynamic DE based navigational strategy

Parameters	Range of parameters
Population Size	30
Dimension of population member	3
Boundary for crossover rate (CR_i)	CR_{min}=0.1, CR_{max}=0.9
Boundary for scaling factor (F_i)	{F_{min}=0.4, F_{max}=1.4} or {F_{min}=-1.4, F_{max}=-0.4}
Initial values of control parameters	F_1=0.5 and CR_1=0.5
Maximum no. of iterations (i_{max})	200

To prevent any possibility of collision with obstacles, the boundary for generated population has also been defined in eq. 5.14 of Section 5.4 which has been strictly followed here. Initial values of control parameters for DDE based path planning algorithm have been chosen empirically as recorded in Table 6.1.

Boundaries for control parameters (F_i and CR_i) are of great importance as they directly affect the convergence behaviour of proposed dynamic DE based optimization. The outcome of mutation process in DDE algorithm can be regulated based on value and sign of F_i as defined in Section 6.4. To adjust the direction of underwater robot's motion, the boundaries for F_i must be decided based on robot's current position within given scenario such as: For $\{robx_k, roby_k, robz_k\} > \{nobx, noby, nobz\} > \{goalx, goaly, goalz\}$, F_i will be limited in $\{F_{min}=0.4, F_{max}=1.4\}$. When robot has to move in reverse direction as $\{goalx,goaly,goalz\} > \{nobx,noby,nobz\} > \{robx_k, roby_k, robz_k\}$, then F_i will follow the limit $\{F_{min}=-1.4, F_{max}=-0.4\}$.

CR_i will always be positive and limited within the range given in Table 6.1. Searching for next best possible pose of robot may continue for maximum 200 iterations (i_{max}) as considered in Table 6.1. But if best fitted population vectors from two consecutive iterations are almost same, then iterations will be stopped as stated in Figure 5.

Next, fitness of randomly generated population members have been evaluated by following eq. 6.7 which is the second step of proposed DDE based optimization (as given in Section 6.4). Two major objectives of navigation, safe avoidance from obstacle and minimization of path length have been considered while designing fitness function for three-dimensional path planning algorithm in Section 5.4.1. Fitness function for DDE based path planning algorithm has been chosen similar as eq. 5.21 of previous chapter.

According to flowchart of Figure 6.3, subsequent steps of path optimization have been performed by following the steps of proposed DDE as narrated in Section 6.4. After completion of all required iterations of DDE, the best fitted vector of current population will be confirmed as next position for robot. Underwater robot will then take a proper turn along with linear motion to reach that position which is at a safe clearance from obstacle. The detail about change in heading angle has already been described by eq. 5.16 and eq. 5.17 of Section 5.4.The process will be repeated for until the underwater robot reaches the goal position after avoiding all obstacles within given scenario.

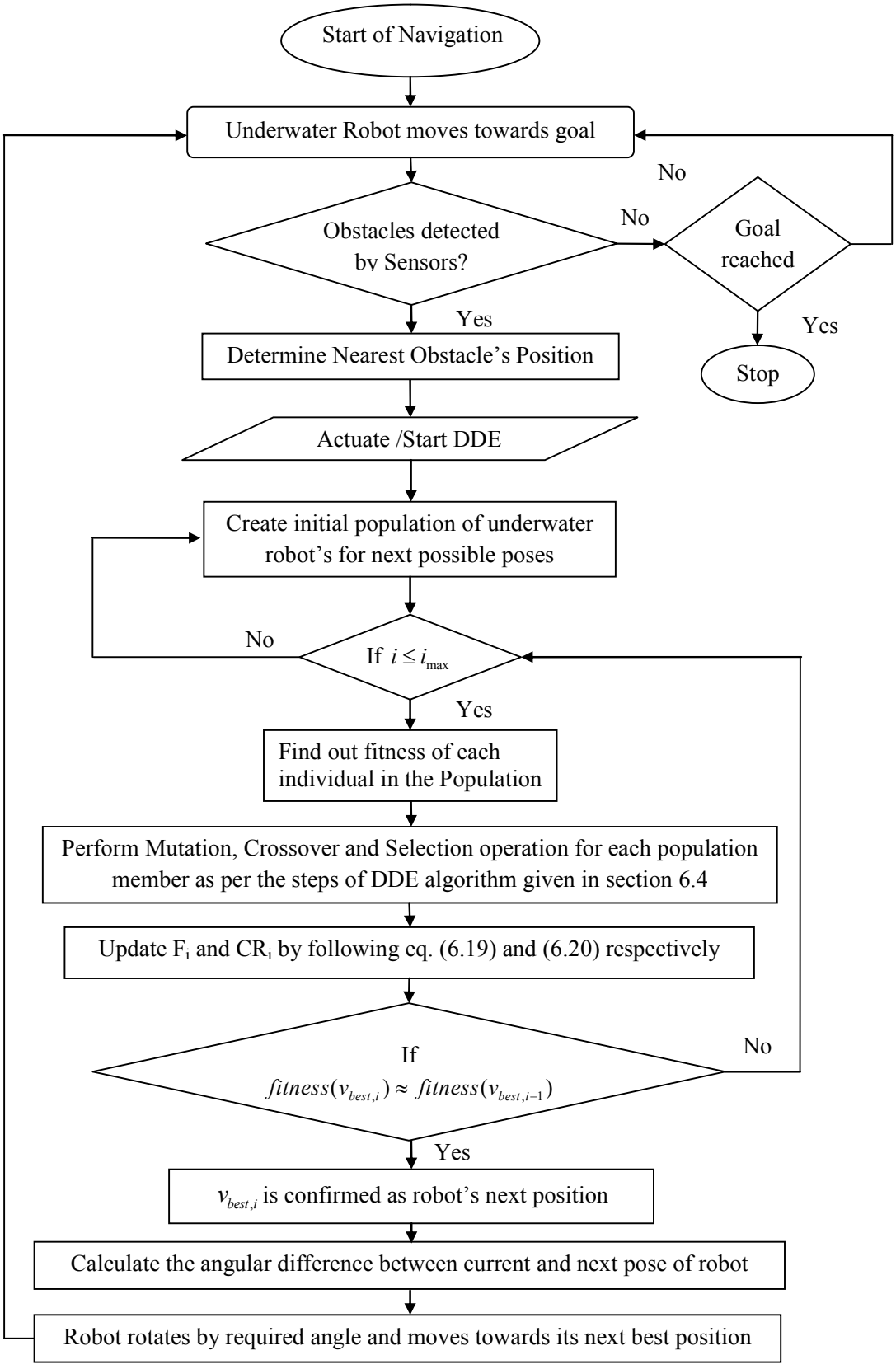


Figure 6.3 Flow chart for path planning based on DDE algorithm

6.6 Simulation Study

Proposed DDE based optimization method has been implemented as navigational strategy of robot in MATLAB based three-dimensional world where obstacles of different shapes and sizes are distributed in a cluttered manner. The objective of simulation study is to achieve collision free near optimal path for underwater robot by employing the proposed DDE based path planning algorithm. The convergence speed and obstacle avoidance ability of proposed algorithm has been verified for navigation within simulated environment whether it is satisfactory or not? Comparison with other three-dimensional navigational approaches has been executed to ensure the feasibility of proposed Dynamic DE based navigational strategy. Assumptions for performing three-dimensional navigation within MATLAB based simulated environment have already been discussed in Section 4.5 of Chapter 4.

6.6.1 Performance analysis of DDE with respect to other versions of DE

A simple simulated environment with only one obstacle (Figure 6.4) has been considered here, where robot has started from (107,70,0) and progresses towards the target point (653,660,90). An elliptical shape obstacle which is associated with maximum dimension of 60 units has been located at (436, 390, 32) in between start and goal point. So, the navigational controller of robot has to choose a path which must be at safe distance from specified obstacle. The scenario given in Figure 6.4 has been used to compare the convergence behaviour of proposed Dynamic DE algorithm with five other metaheuristics such as GA [124], *j*DE [183], JADE [184], SADE [185] and SspDE [186]. Control parameters used for all algorithms have been specified in Table 6.2 by following assumptions of corresponding researchers. For comparison purpose, values of few parameters have been chosen same for all navigational strategies as per the specifications of three-dimensional path planning problem such as, Size of population: 30, Dimension of solution vector:3 and Maximum permissible iterations (i^{max}): 200.

Each individual algorithm has been employed as obstacle avoidance strategy while tracing path from start to goal within the simulation scenario of Figure 6.4 in separate runs. On detection of obstacle at (360, 310, 20) which is at a safe clearance (more than obstacle's radius) from obstacle, the assigned optimization method has been actuated to search the next best position. Next, the underwater robot has moved towards that estimated position and the process has been repeated up to (500,430,40) where influence

of obstacle has been ended (Figure 6.4). After that robot has headed towards goal point as no further obstacle has been detected. For given situation, average convergence curve (average best fitness value vs. iterations) for each optimization algorithm has been shown in Figure 6.5 as an average of their 30 runs.

Table 6.2 Control parameter values for GA and adaptive DEs used in simulation

Algorithms	**Details of control parameters**
GA[124]:	Crossover rate: 0.9; Mutation rate: 0.01
jDE [183]:	Mutation strategy: DE/rand/1/bin and DE/current to best/1/bin; Upper and lower limit of F ϵ (0.1,1): {F_l=0.1, F_u=0.9}; CR ϵ (0,1) and Probabilities to adjust F and CR: τ_1, τ_2 = 0.1 respectively
SADE [185]:	A pool of mutation strategies: DE/rand/1/bin, DE/rand to best/2/bin, DE/rand/2/bin, DE/current to rand/1; {F = N(0.5,0.3) ϵ (-0.4,1.4)}; {CR = N(CR_m, Std) ϵ (0,1); where, CR_m= 0.5, Std=0.1}; Learning Period (LP) = 20 iterations
JADE [184]:	Mutation strategy: DE/current-to-pbest/1 (with and without archive); F_i = $randc_i(\mu_F, 0.1)$, $\mu_F$is location parameter of Cauchy distribution and initialized as 0.5; Similarly, CR_i = $randn_i(\mu_{CR}, 0.1)$, $\mu_{CR}$is mean and initialized as 0.5; c: Constant in Cauchy distribution that controls adaptation rate of F and CR (c=0.1 normally); *p*: Greediness of mutation strategy (*p*=0.05)
SspDE[186]:	A pool of mutation strategies: DE/rand/1/bin, DE/rand to best/2/bin, DE/rand/2/bin, DE/current to rand/1; F ϵ (0.1,1); CR ϵ (0,1); Learning Period (LP) = 20 iterations and Reassignment Probability (RP)=0.8
Proposed DDE:	Mutation strategies and control parameters are used as per given details in Table 6.1.

In GA [124], population members with higher fitness are normally selected as parent vectors to perform crossover. After crossover, generated new solution with better fitness has been taken into account and others are discarded. Therefore, within few generations, diversity in population may be drastically reduced resulting in local minima situation as shown in Average fitness curve of GA (Figure 6.5) for given scenario (Figure 6.4). Mutation process may inject some exploration. Only one offspring is generated per iteration which makes the convergence speed slow and also increases computational time. There may be nonlinear increase in time for sorting of population when population size is huge for GA.

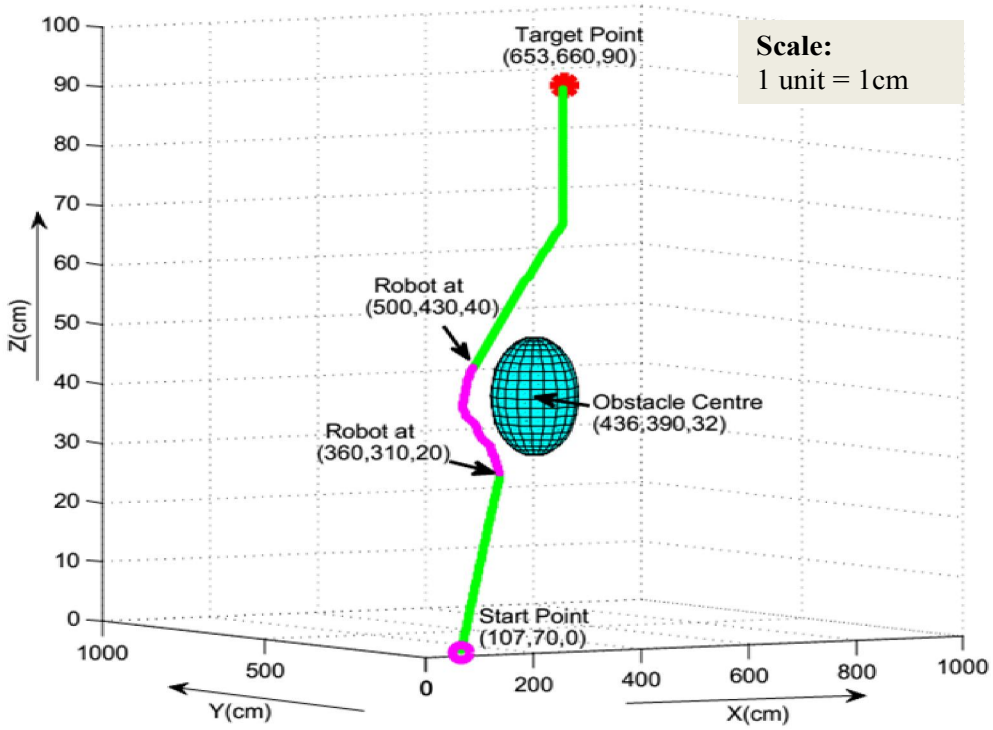


Figure 6.4 Simulation environment with single obstacle

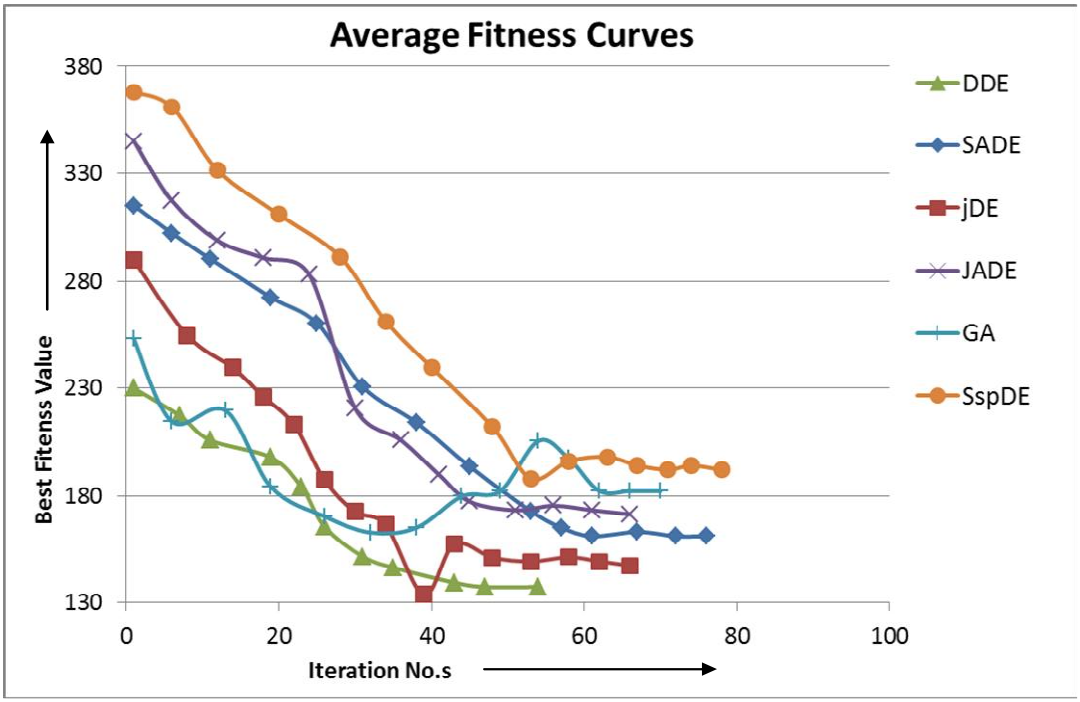


Figure 6.5 Average fitness curves for six algorithms for avoiding obstacle within simulation scenario of Figure 6.4

Self-adaptation process of SaDE [185] requires sufficient learning period initially to gain experiences about performances of different sets of control parameters before assigning proper parameter values with specified mutation strategies. So, including initial learning time, comparatively large iteration numbers are needed for convergence of SaDE [185] as shown in Figure 6.5. For three-dimensional path planning problem, SaDE [185] has performed with slower convergence speed but has shown better global search ability than GA [124].

Similarly, another self-adaptive version of DE, SspDE [186] also requires learning period to assess the list of control parameters and mutation strategies associated with each population members during selection process of DE. After completion of learning period, control parameters and mutation strategies from the wining lists will be reassigned to the population members with respect to the acceptance probability of newly generated solution vectors. In spite of sufficient diversity in solutions and well-tuned control parameters, performance of SspDE [186] has found to be less reasonable than SaDE [185] regarding convergence speed as shown in Figure 6.5.

Avoiding premature convergence of "DE/current to best/1" strategy, JADE [184] has randomly chosen 100p% ($p \in (0,1)$) best solutions from random population to generate trial vectors. Use of multiple best solutions and voluntary external archive of previous poorer solutions in mutation scheme maintains good diversity in population as well as fast convergence speed as shown in convergence curve of Figure 6.5.

The performance of *j*DE [183] in given scenario of path planning has found to be more impressive than other versions of DE. The mutation scheme employed in jDE [183] provides faster convergence speed towards best solution than others which may result in local minima situation as shown in Figure 6.4. As comparison of convergence behaviours has been performed in Figure 6.4, so, it can be seen that proposed DDE algorithm has been converged more quickly, without being stuck in local minima, along with lowest fitness value than GA and other versions of DE.

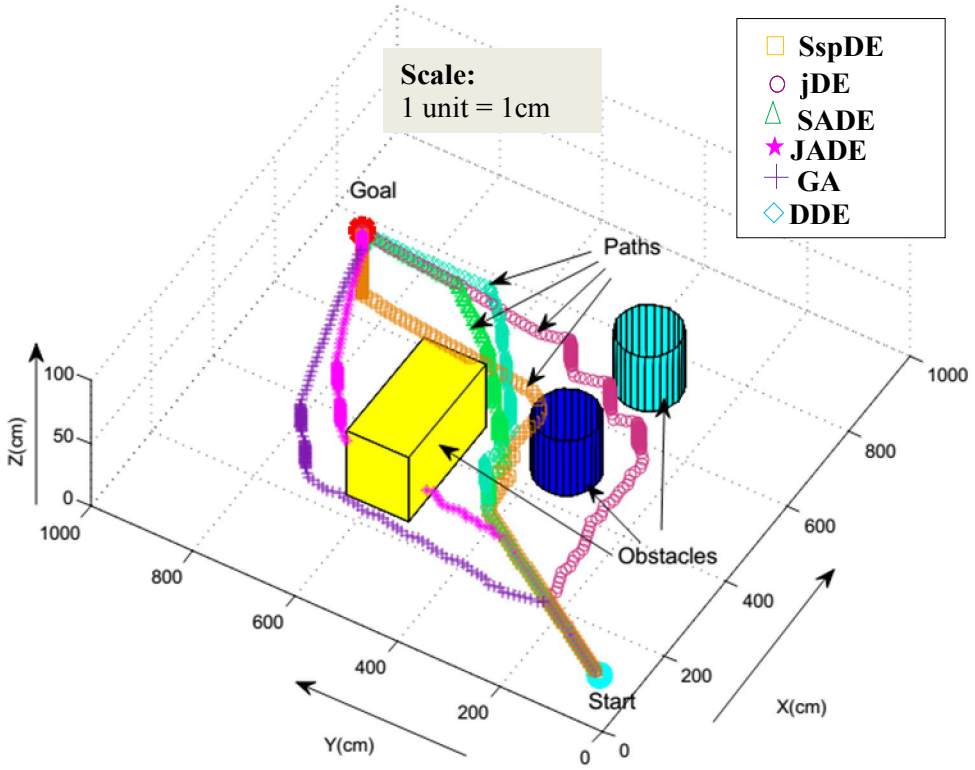


Figure 6.6 Simulation paths traced by six navigational strategies

Apart from convergence behaviour, navigational efficiencies of above mentioned six metaheuristics have also been evaluated here. For that purpose, one more simulation scenario associated with multiple obstacles has been considered in Figure 6.6. Comparison of above mentioned six algorithms has been executed with respect to the deviation in path length from its optimum path, total path length to be travelled and also clearance from obstacle. For given scenario, the straight distance between the start and goal points without considering obstacles (860.18 cm) has been considered as optimum path length. Table 6.3 contains details about actual path lengths, percentage deviations from optimum path and observations on collision avoidance for GA [124], *j*DE [183], JADE [184], SADE [185], SspDE [186] and proposed DDE, which have been already applied in Figure 6.6 as navigational strategies.

Analysing performance of above mentioned algorithms, proposed dynamic differential evolution and jDE [183] have found to be most effective for safe avoidance from obstacles and path length minimization. For every algorithm, the simulation has been repeated about 12 times and feasible path among them has shown in Figure 6.6. SaDE

[185] and JADE [184] have traced paths with almost same path length but JADE [184] has failed to avoid collision. Performances of GA [124] and SspDE [186] in current scenario are found to be unsatisfactory. Present comparison has proven the dominance of proposed DDE algorithm over considered metaheuristics up to a certain extent.

Table 6.3 Comparison of proposed DDE with GA and other versions of DE regarding path length and obstacle avoidance behaviour for simulation scenario of Figure 6.6

Path planning algorithms	Deviation from optimum path (in %)	Path length (in cm)	Safe from collision (Yes/No)
GA[124]	17.27	1039.7	No
jDE[183]	12.91	987.6	Yes
SaDE[185]	14.14	1001.8	Yes
JADE[184]	13.89	998.9	No
SspDE [186]	15.81	1021.7	No
Proposed DDE	5.78	912.9	Yes

6.6.2 Simulation results of DDE for robotic behaviors

Proposed Dynamic DE has been successfully implemented as navigational strategy of underwater robot in three-dimensional simulation scenarios (Figures 6.7 - 6.8) to perform robotic behaviours such as obstacle avoidance, wall following and target seeking etc. during navigation. Whenever the robot is facing obstacle arrangement like 'U' or 'L' shape on its way to goal, the major intent of navigational controller will be to make free the robot from such dead end condition (as shown in Figure 6.7). To achieve this, robot has to change its heading direction irrespective of the goal position of given scenario. By obeying wall following behaviour, path parallel to the obstacle has been traced by the proposed DDE based navigational strategy in Figure 6.7. Once robot is able to recover itself from the influence of trap like situation, obstacle avoidance and target seeking behaviours of DDE based path planner have been activated in a cooperative manner.

Simulation scenario of Figure 6.8 contains a large number of static obstacles distributed in a random way. Proposed DDE approach based navigation has been found to be very much effective for tracing collision free near optimal path in given scenario of Figure 6.8. In all simulation scenarios, DDE based path planner has shown desirable performance by keeping safe clearance from obstacles and also minimizing the path length to some extent during navigation. Best among 20 runs of each simulation result has been exhibited here in Figures 6.7 and 6.8 respectively.

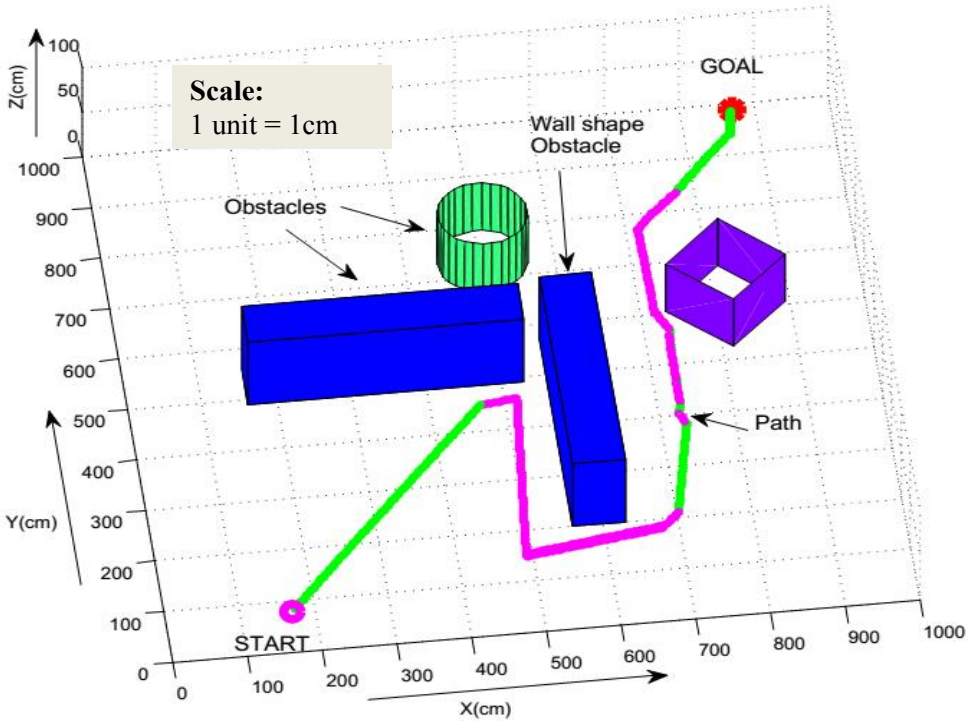


Figure 6.7: Simulation result for wall following behaviour

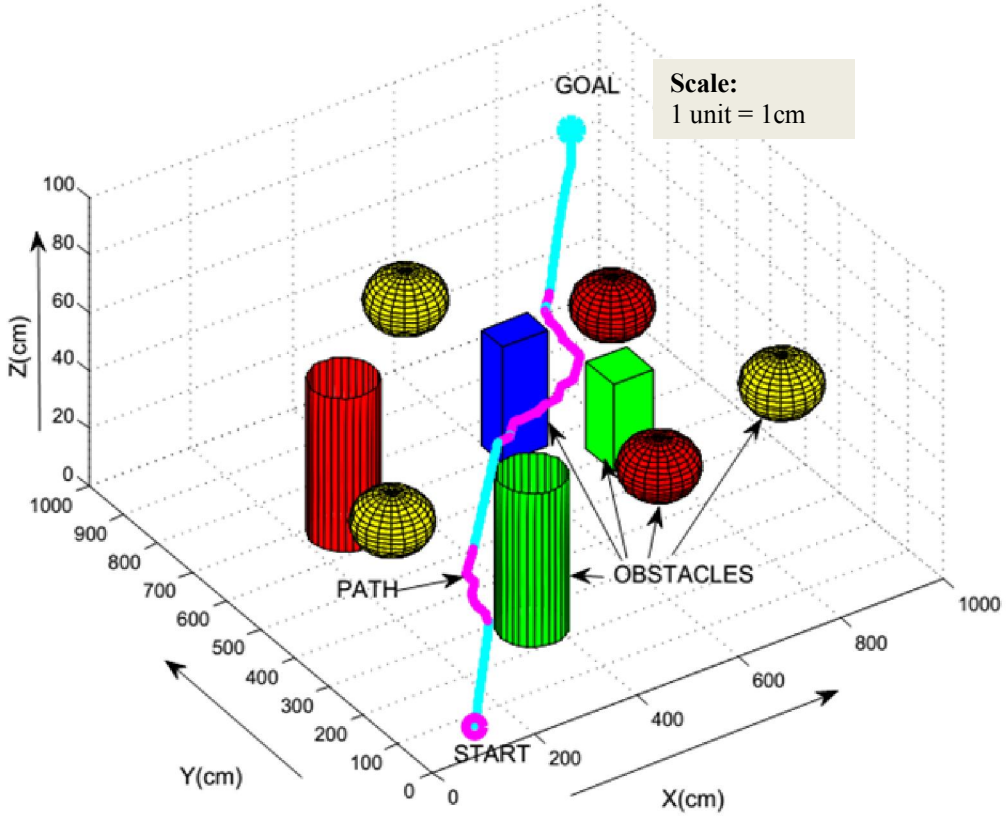


Figure 6.8: Simulation result for Obstacle avoidance and Target seeking behaviour

6.6.3 Comparison with other three-dimensional navigational approaches

Authentication of proposed DDE approach based navigational strategy has been ensured by performing comparison with other popular three-dimensional navigational approaches like fuzzy logic and heuristic potential field approach. To get justification in comparative study, the simulation scenarios along with the obstacle arrangement (Figures 6.9 (a) and 6.10(a)) as used by pervious researchers have been reproduced here through MATLAB based simulations (Figures 6.9 (b) and 6.10(b)). DDE based path planning algorithm has been successfully employed in each newly designed simulation scenario to achieve collision free path. The details of comparative study have been given as follows:

(a) Jiang and Zhu [187] have implemented fuzzy based navigation strategy for AUV without considering complete map of environment. Acceleration of left and right thrusters are the outputs of FLC for input variables such as left, front and right sensor's value, heading angle and surge velocity. Combining horizontal and vertical path, synthetic three-dimensional path has been generated in simulation mode (Figure 6.9 (a)) while avoiding static obstacles. Static MATLAB based scenario (12m X 12m X 12m) contains six obstacles with different shape and sizes in between start (0,0,0) and end point (10,10,10) (Figure 6.9 (a)).

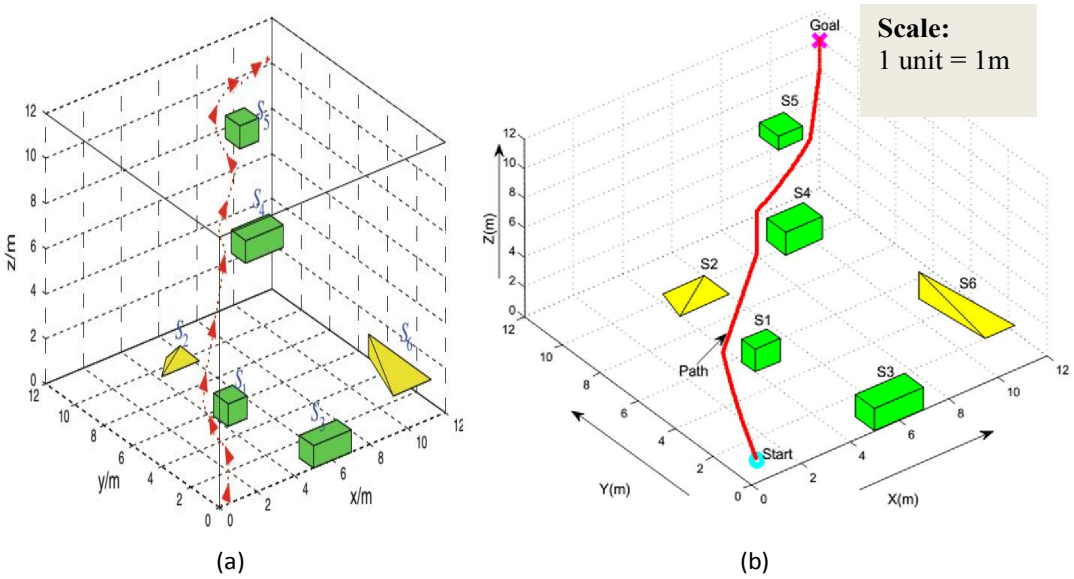


Figure 6.9 (a) Three-dimensional static path planning for AUV using fuzzy logic control by Jiang and Zhu [187]; (b) Navigational path traced by proposed DDE approach based navigational strategy

Proposed DDE algorithm has been implemented in simulated workspace of Figure 6.9 (b) where start and goal positions and obstacles' shape, size and arrangement are same as Figure 6.9 (a). Comparison between two approaches regarding path length has been exhibited in Table 6.4.

(b) Miao and Huang [25] have proposed a novel potential field approach for three-dimensional navigation by incorporating heuristic obstacle avoidance scheme. Heuristic potential field (HPF) approach has drawn the realistic path for AUV in simulated workspace associated with three unknown static obstacles (Figure 6.10(a)). HPF approach has also been proven by Miao and Huang [25] to be more efficient than SA and GA regarding online processing time. Simulation scenario of Figure 6.10(b) which contains obstacle arrangement similar as Figure 6.10(a) has been used to show performance of proposed DDE algorithm. More clearance from obstacles has been observed in path traced by DDE based navigation (Figure 6.10(b)) than HPF [25] in Figure 6.910(a).

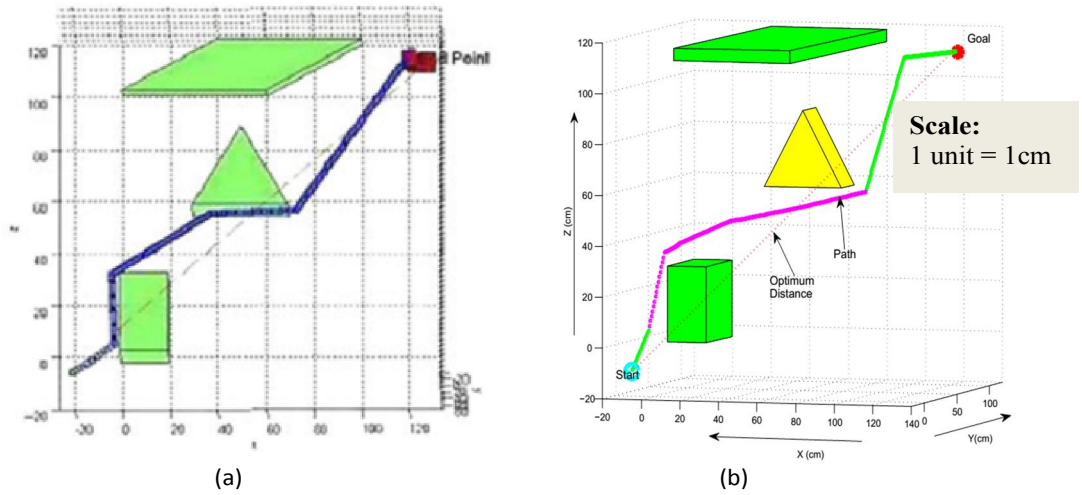


Figure 6.10 (a) Simulation result using Heuristic Potential Field as shown by Miao and Huang [25]; (b) View of Path obtained by implementing proposed DDE based navigational approach within obstacle arrangement same as Figure 6.10 (a)

The red dotted line connecting start and target point has been treated as the shortest path length that underwater robot should follow in absence of obstacles. Actual length of paths as traced by heuristic potential field [25] and proposed DDE based navigational approaches have been listed in Table 6.4 respectively. In both

comparisons, proposed DDE based navigational strategy has traced shorter path length than other considered navigational approaches as recorded in Table 6.4. Therefore, proposed DDE approach can be preferred as three-dimensional navigational strategy over other approaches.

Table 6.4 Comparison between Proposed DDE algorithm and other navigational strategies for navigational performance within simulated environments

Figure No.	Navigational Strategy	Length of 3D path traced by underwater robot (in cm)	Deviation (in %)
6.9(a)	Fuzzy logic based approach [187]	223.9	3.69
6.9(b)	Proposed Dynamic DE based approach	215.93	
6.10(a)	Heuristic potential field approach [25]	252.85	2.3
6.10(b)	Proposed Dynamic DE based approach	247.17	

6.7 Experimental Results

Small size underwater robot "GNOM baby" has been used to perform experimental verifications of simulated performance of proposed Dynamic DE based path planner. The detail specifications of "GNOM baby" has been discussed in Appendix-A. In experimental mode, submerged obstacles of different shapes and sizes have been considered as static by nature. While arranging locations for obstacles, start and goal points within experimental environment, a sincere attempt has been taken to follow the simulation scenario of Figure 6.8. Controller of underwater robot embedded with proposed dynamic DE based navigational strategy has found to be successful for avoiding obstacles within experimental scenario of Figure 6.11. In the way from start to goal, underwater robot has followed the global best positions chosen by proposed DDE based optimization method in a sequential manner to trace a collision-free near optimal path during experiment. Figure 6.11 shows the positions of underwater robot GNOM-baby at different intermediate stages of navigation within real time underwater environment. Before the navigation has started, the complete view of experimental environment with obstacles and goal point can be viewed at stage 1.

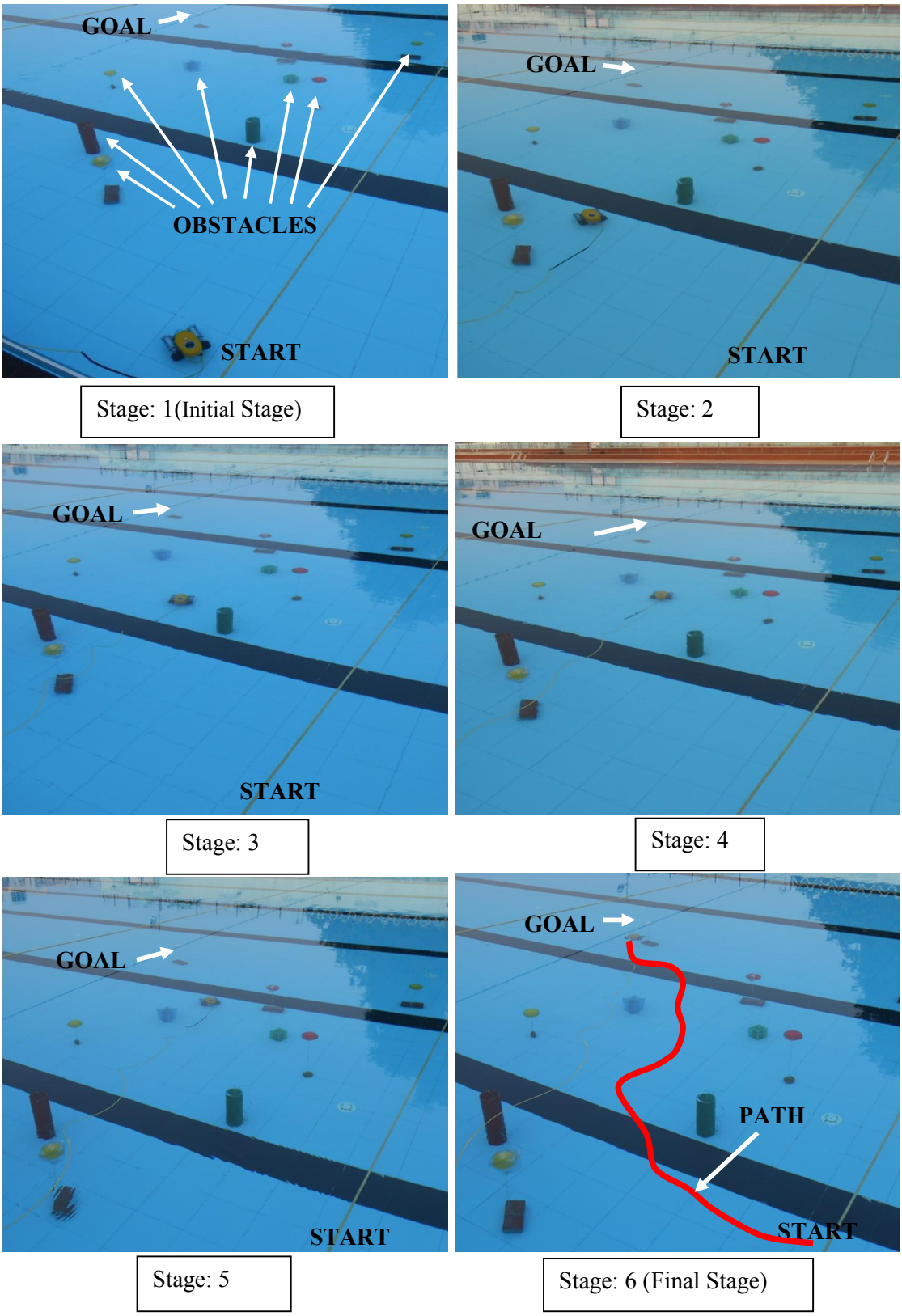


Figure 6.11: Experimental view for Dynamic DE based navigational strategy with obstacle as per the simulation result of Figure 6.7

After the initiation of motion, one possible location of GNOM-Baby has been shown in stage 2. A number of obstacles have been avoided by underwater robot during navigation as exhibited in few stages (from stage 3 to 5) of Figure 6.11. Underwater has successfully reached its destination at stage 6 of Figure 6.11.

6. 6.1 Comparison between simulation and experimental results:

The performance of proposed Dynamic DE based navigational approach can be verified by comparing simulation and experimental results. The best solution of any population based evolutionary approach may differ in separate runs. To ensure the optimization ability of proposed dynamic DE based approach, execution of simulation (Figure 6.8) and real time experiment (Figure 6.11) has been repeated for 20 times within same scenario. Path length and travel time for each run of path planning algorithm in simulated and experimental modes have been noted in Table 6.5 and 6.6 respectively.

Table 6.5 Comparison between experimental and simulated results for path length in Scenario1 (Figure 6.8 and 6.11)

1	2	3	4	5	6
No. of Runs	**Path length in simulated mode (in cm)**	**Deviation from Optimum path (%)**	**Path length in experimental mode (in cm)**	**Deviation from Optimum path (%)**	**Difference between column 2 and 4 (%)**
1st	1157.2	4.60	1224.1	5.40	5.78
2nd	1167.1	5.50	1219.8	5.04	4.51
3rd	1161.1	4.96	1227.7	5.71	5.73
4th	1170.4	5.79	1235.7	6.40	5.58
5th	1165.7	5.37	1223.2	5.33	4.93
6th	1163.9	5.21	1232.6	6.13	5.90
7th	1160.4	4.89	1229.2	5.84	5.92
8th	1158.5	4.72	1226.1	5.57	5.82
9th	1162.4	5.07	1230.3	5.93	5.84
10th	1155.5	**4.45**	1224.1	5.40	5.93
11th	1164.2	5.23	1217.4	**4.83**	4.57
12th	1161.2	4.96	1229.9	5.90	5.92
13th	1165.1	5.31	1235.5	6.39	6.05
14th	1163.5	5.17	1228.1	5.75	5.55
15th	1171.5	5.89	1236.1	6.44	5.52
16th	1159.7	4.83	1227.5	5.70	5.84

17th	1164.8	5.29	1233.4	6.21	5.89
18th	1162.1	5.04	1228.7	5.80	5.73
19th	1168.6	5.63	1237.1	6.53	5.87
20th	1160.8	4.93	1236.9	6.50	6.55
Average difference in path length between simulated and experimental results					5.67

The diagonal distance between start and goal point of given scenario or optimum path length has been recorded as 1106.3 cm and 1161.4 cm for simulation (Figure 6.8) and experimental (Figure 6.11) results respectively. Differences between actual and optimal path lengths have also been estimated for both simulation and experimental results as depicted in Table 6.5. Minimum deviations from optimal path length have been recorded in 10th and 11th run of proposed DDE based path planner for simulation and experimental results respectively. The average difference between simulated and experimental result regarding path length and time taken has been computed to validate the feasibility and robustness of proposed Dynamic DE based navigational strategy.

Table 6.6 Comparison between experimental and simulated results for travel time in Scenario1 (Figure 6.8 and 6.11)

1	**2**	**3**	**4**
No. of Runs	**Travel time in simulated mode (in sec)**	**Travel time in experimental mode (in sec)**	**Difference between column 2 and 4 (in %)**
1st	18.37	19.13	4.13
2nd	25.37	26.69	5.20
3rd	17.86	18.80	5.25
4th	22.95	24.37	6.20
5th	14.76	15.37	4.14
6th	23.75	25.04	5.40
7th	21.10	22.39	6.12
8th	15.87	16.82	5.97
9th	22.35	23.57	5.43
10th	12.98	13.71	5.57
11th	15.12	15.94	5.43
12th	13.05	13.87	6.27
13th	15.33	16.19	5.63
14th	13.37	14.07	5.19
15th	14.83	15.62	5.35
16th	13.18	13.93	5.69
17th	17.39	18.65	7.28

18th	20.39	21.51	5.49
19th	13.75	14.53	5.70
20th	13.19	14.04	6.39
Average difference in travel time between simulated and experimental results			5.59

In three-dimensional workspace, different obstacle arrangements have been made to execute a number of simulations and their respective experimental verifications by employing proposed DDE based navigational strategy. Comparisons between simulation and experimental results have been carried out for five different scenarios. Outcome of broad comparative study with respect to path length and travel time has been recorded in Table 6.7 and 6.8 respectively. As negligible errors have been found in all comparisons, the navigational performance of proposed dynamic DE based navigational strategy can be treated as satisfactory.

Table 6.7 Comparisons between experimental and simulated results for five more scenarios regarding path length

Different Scenarios	Best path length in Simulation (in cm)	Best path length during Experiment (in cm)	Difference (in %)
Scenario-2	827.9	876.1	5.81
Scenario-3	783.3	827.7	5.68
Scenario-4	996.4	1055.4	5.93
Scenario-5	1319.6	1386.3	5.06
Scenario-6	1143.3	1210.6	5.89

Table 6.8 Comparisons between experimental and simulated results for five more scenarios regarding travel time

Different Scenarios	Travel time during simulation (in sec)	Travel time during experiment (in sec)	Difference (in %)
Scenario-2	9.41	9.89	5.14
Scenario-3	8.61	9.06	5.22
Scenario-4	10.49	11.10	5.81
Scenario-5	14.83	15.71	5.95
Scenario-6	12.56	13.23	5.31

6.8 Summary

In this chapter, a dynamic version of Differential Evolution has been formulated and employed as three-dimensional path planning algorithm. From recent investigation, major features of proposed DDE based navigation can be summarized as follows:

- A self-learning approach has been proposed for choosing the mutation strategy in DE algorithm based on few factors such as success rate of mutated vector in crossover operation, fitness value of updated population and iteration number. Such adaptive mutation scheme has effectively performed global search during early iterations and local search near about global optima at final stage of optimization.

- Control parameters of DE have been dynamically tuned throughout iterations based on fitness value of recent population and iteration number to balance the exploration and exploitation abilities of search process in a cooperative manner.

- Proposed Dynamic Differential Evolution based optimization has stochastically found out the next global best position for underwater robot in each iteration while maintaining safe clearance from obstacles.

- Convergence behaviour of proposed algorithm has been compared with genetic algorithm and other available versions of DE for present path optimization problem. Comparison has certified the superiority of proposed DDE over other methods regarding convergence speed and path optimization ability.

- Comparison of proposed method with other underwater navigational strategies in simulation mode endorses the practicality of current algorithm.

- Basic reactive behaviours of underwater robot have been verified for proposed Dynamic DE based navigation in both simulation and experimental results. Performance assessment of proposed DDE based three-dimensional navigational strategy has been done by comparing simulation and experimental results. Average error between simulated and experimental results has been found to be 5.67% for path length and 5.51% for travel time.

Easy implementation, minimum computational complexity and faster convergence speed of proposed DDE has been found to be suitable enough for employing it as obstacle

avoidance and path optimization strategy of underwater robot. In next chapter, one more stochastic approach with less mathematical computation has been employed as navigational strategy of underwater robot.

7. Navigation of Underwater Robot based on Adaptively Tuned Harmony Search Algorithm

Evolutionary approaches have received good response while solving path planning problem for underwater robot in previous chapters. In this chapter, one unconventional metaheuristic approach which relies on perfect analogy between natural music improvisation process and optimization technique has been utilized to find out collision-free near optimal three-dimensional path for underwater robot. Major challenge of this chapter will be the balance between diversification and intensification during path optimization process.

7.1 Introduction

Three-dimensional path planning problem can be solved by determining the best fitted locations for underwater robot in a sequential manner while navigating from start to goal point of given scenario. Iterative optimization ability of population based stochastic search has been found to be enough competent for resolving such localization problem of underwater robot. This chapter has suggested a new navigational strategy for underwater robot based on Harmony search (HS) algorithm which imitates musician's search process for pleasing harmony as introduced by Geem et al. [141]. Optimization behavior of Harmony search integrates certain steps of some popular heuristic approaches such as Tabu Search, Simulated Annealing and GA. Fewer mathematical computations and no particular initial values for decision variables make HS preferable over other metaheuristics to perform online optimization for large scale engineering problems [129]. Generation of new vector considering all harmonies of memory enriches the exploration ability of HS whereas GA produces offspring from parent vectors only. Moreover, HS has shown better search quality and faster evolution speed than GA in numerical comparisons as HS can adjust each decision variable of solution vectors individually [138-140].

In the present study, HS's ability of finding high performance region of search space within feasible time has been employed for finding collision free near optimal path of underwater robot. Though HS has shown good exploration ability during global search, but random perturbation of solution vectors in evolution strategy may adversely affect the local search ability by reducing convergence speed [188]. Several improved versions of HS have already been introduced and used to solve diverse real-time optimization

problems [189-191] successfully. But effectiveness of HS's variants solely depends on the choice of control parameters and may also vary with the dimension of given problem. Constant values of control parameters within a specific range may not assure the optimal performance of the Harmony search. Therefore, a new adaptive tuning process has been proposed here to automatically adjust the control parameters of HS at different stages of search. To balance the randomness and exactness in proposed version of HS, vector perturbation strategy has been varied dynamically during pitch adjustment process. Robustness of proposed Adaptively Tuned Harmony Search as a navigational strategy has been verified in simulation and experimental studies on underwater navigation.

7.2 Basic Harmony Search and its modified versions

Harmony in music may represent the solution vectors with certain dimension for a nonlinear optimization problem and music improvisation process may denote local and global search processes in a combined manner [192]. As a first step of HS algorithm, harmony memory of specified size (HMS) accumulates randomly generated harmonies or solution vectors $q^m = \{q_1^m, q_2^m, ..., q_v^m\}$ limited within lower bound and upper bound of respective dimension:

$$q_d^m = q_d^{\min} + rand(0,1) \times (q_d^{\max} - q_d^{\min}) \qquad (7.1)$$

Where, m = 1, 2, …., HMS: number of solution vectors, 'd': the dimension of each vector and $\{q_d^m, q_d^{\min}\}$: Boundary of variables.

Based on harmony memory consideration rate (HMCR) and the pitch adjusting rate (PAR) value, HS performs a stochastic random search within harmony memory to generate only one new harmony vector which must be tuned by bandwidth (bw) value. (1-HMCR) is the probability of random selection of solution vector from outside of harmony memory but within the specified limit of search space. The improvisation process of HS [130] has been illustrated here:

For each $q_m; \{m \in (1, M)\}$ do

If $HMCR \geq U(0,1)$ %memory consideration

Then $q' = q_v^m$, where v ~ U (1,….., V) (7.2)

if $PAR \geq U(0,1)$ %pitch adjustment

then $q'' = q' \pm rand(0,1) \times bw$ (7.3)

Where bw: Bandwidth value bounded within a certain range

end if

Else /* random selection */

$q' = q^{\min} + rand(0,1) \times (q^{\max} - q^{\min})$ (7.4)

End if

End for

If the fitness of new vector is better than existed worst member, it will replace that worst one by modifying the Harmony Memory (HM). This procedure will be iterated until some explicit stopping norms are gratified.

HMCR probability determines whether the solution vector will be chosen from HM or will be generated by random selection (eq. 7.2). With the progress in iterations, linearly increasing HMCR value from 0.9 to 1 may help to endow the required diversity in harmony memory initially and fine tuning of solution vectors at final stage [138]. Solution vector chosen with HMCR probability has been further adjusted based on PAR probability in eq. 7.3. Control on probabilistic value of PAR may improve the exploitation ability of HS. Tuning of newly generated vector within specified bandwidth (bw) value may facilitate both global and local search. In early generations, a large value of PAR and bw may enforce increment of diversity in harmony memory. Both PAR and bw should be decreased gradually to fine-tune the solution vectors at final stage of iterations. The random selection in HS allows the exploration of new regions of search space.

Therefore, constant value of PAR and bw may limit the optimization ability of HS algorithm which has been tried to be overpowered in IHS algorithm [130] where PAR and bw can vary dynamically with iteration number. Omran and Mahadavi [131] have incorporated the concept of PSO model in HS by employing global best value of solution vectors instead of bandwidth based adjustment. Global harmony search (GHS) may face the premature convergence as it considers only best fitted solution. Pan et al. [132] have modified GHS algorithm by adding an adaptive learning mechanism for control parameters. In last few years, several improved versions of Harmony search have been introduced by researchers based on different adaptation mechanisms for HS parameters. Many of them have proven to be advantageous for real world optimization problems [137,139-140]. All HS variants have shown their optimization ability in various case

studies on different applications [129]. But their feasibility and efficiency have not been verified for the problems with large dimensions.

In spite of good global search ability, HS may converge slowly or may trap into local minima because of three main reasons: (a) HS performs based on past experiences; (b) HS's new vector generation mechanism does not care about quality or fitness of solution vectors in harmony memory; and (c) Absence of fine tuning ability for the final solution towards the end of the search [188]. To improve both search quality and convergence speed of HS, adaptation of control parameters and pitch adjustment schemes has been proposed in next section based on variation in fitness of population.

7.3 Design of Proposed Adaptive version of Harmony Search Method

Regulation of search behaviour from explorative at the beginning to exploitative at the end of optimization has been considered as major objective of proposed modifications on original Harmony Search. Fine tuning mechanism has been additionally incorporated within the global search process of HS to perturb solution vector near about global optima in an efficient manner. Proposed integration of global and local search abilities in HS with respect to fitness of current population and iteration numbers has been systematically illustrated here along with its Pseudo-Code.

Steps of Adaptively Tuned Harmony Search Method:

Step 1: *Initialization*

Random initialization of harmony memory within specified boundaries can be performed by using eq. 7.5 of Pseudo code. Suitable ranges for control parameters of HS have also been declared at this step. Fitness value of each solution vector of harmony memory will be evaluated by following eq. 7.6 and will be stored in descending order.

Step 2: *Modified Harmony Improvisation Process*

In each iteration, improvisation process has followed three rules: HMCR, PAR and Randomization. Here, pitch adjustment scheme has been modified by varying vector perturbation rule.

Step 2a: Harmony memory consideration

Based on $HMCR_e$ probability only one solution vector will be chosen from harmony memory (as given in eq. 7.7) same as conventional HS algorithm.

Step2b: Modified adaptive pitch adjustment process

Any one of three perturbation schemes (as given in eq. 7.8, 7.9 and 7.10) will be activated in one iteration based on probabilistic value of PAR_e and number of iterations as specified in Pseudo code. At initial phase of evolutions $e \ll e_{max}$, PAR_e has been kept high to explore the search space. In this case, the difference between best and worst fitted vector has been used as perturbation factor in pitch adjustment rule of eq. 7.10 to enhance the diversity in population. As iteration progresses, PAR_e will decrease linearly and probability of choosing perturbation schemes of eq. 7.9 or 7.8 will be high. In the middle phase of evolution ($e < e_{max}$), eq. 7.9 may get activated. Mean value information of current population has been incorporated in adjustment rule to maintain the divergent and convergent behaviour of search process in a coupled manner. Adaptive bandwidth parameter has been used to scale the difference between mean and worst valued solution vectors in eq. 7.9.

At final stage of optimization, $e \approx e_{max}$, fine tuning of solution vector will be essential to reach global optima with high degree of accuracy. Pitch adjustment rule may follow the eq. 7.8which contains adaptive BW_e to slightly perturb the chosen solution vector.

Step2c: Randomization

Solution vectors with (1-HMCR) probability will be generated from outside harmony memory but within specified range of search space by following eq. 7.11.

Step3: *Selection*

If newly generated vector has better fitness than worst fitted solution vector of present population, then, new one will replace the worst one and selection rate will be increased by one. Otherwise, there will be no change in the harmony memory. After completion of one full iteration of HS, replacement rate (RR) within population can be estimated by eq. 7.12.

Step4: *Adaptive adjustment of control parameters*

In early iteration, $HMCR_e$ will be near about $HMCR_{min}$ by following eq. 7.13. So, probability of randomization will be high enough and new vector will be generated by following eq. 7.11 to explore the search space properly. At initial stage, if newly generated vectors are not enough fitted to be included in harmony memory, then RR is expected to be lower than its threshold of (0.2). In that case, PAR_{max} and BW_{max} will be selected as given in eq. 7.16. High value of PAR_e and low iteration number (e) may select eq. 7.10 as pitch adjustment scheme to enhance the diversity in population. After small number of iterations, both $HMCR_e$ and RR may get increased. In that case, PAR_e will be controlled based on fitness differences between two consecutive iterations in eq. 7.14 and BW_e will be regulated by iteration number, value of RR and both maximum and minimum fitness values of recent population in eq. 7.15. Updated control parameters (PAR_e and BW_e) and perturbation scheme of eq. 7.9 will help to perform a trade-off between diversification and intensification in middle stage of search process. At final stage of search process, $HMCR_e$ will be enhanced up to its maximum and both PAR_e and BW_e will be decreased towards their minimum values. So, fine tuning of solution vectors will be performed in eq. 7.8 to enhance convergence speed.

Steps 2, 3 and 4 will be repeated until the termination criterion (maximum iteration) has been satisfied.

Pseudo Code for Adaptively Tuned Harmony Search:

/* Initialization */

Initialization of control parameters: HMS = M, $HMCR_{min}$, $HMCR_{max}$, PAR_{min}, PAR_{max}, BW_{max}, BW_{min}, Maximum iteration number (e_{max}), Number of decision variables = V, Boundaries for respective decision variables: Minimum (Min_v) and Maximum (Max_v) values.

Initialize Solution Vectors in Harmony Memory:

For m=1: M

For v=1: V

$$q_{v,m} = LB_{v,m} + rand(0,1) \bullet (Max_{v,m} - Min_{v,m}) \quad (7.5)$$

End for

End for

For e=1: e_{max} /* Iteration started*/

For m=1: M /* Fitness Evaluation of each solution vectors*/

Find out $f_{m,e}$ =fitness ($q_{m,e}$) (7.6)

End for

If $U(0,1) \le HMCR_e$ /* Modified Improvisation Process */

/* One solution vector has been chosen from Harmony Memory */

$$q'_e = q_{m,e} \quad (7.7)$$

For v=1: V

If $U(0,1) \le PAR_e$ /* Pitch Adjustment Process */

If $U(0,1) \ge \frac{e_{\max} - e}{e_{\max}}$

$$q''_{v,e} = q'_{v,e} + rand(0,1) \bullet BW_{v,e} \quad (7.8)$$

Else $q''_{v,e} = q'_{v,e} + rand(0,1) \bullet BW_{v,e} \bullet (q_{mean,v,e} - q_{worst,v,e})$ (7.9)

End if

Else $q''_{v,e} = q'_{v,e} + rand(0,1) \bullet (q_{best,v,e} - q_{worst,v,e})$ (7.10)

End if

End for

Else

For v=1: V

$q_{v,e}^{new} = q_{m,v,e}^{\min} + rand(0,1) g (q_{v,e}^{\max} - q_{v,e}^{\min})$ /* Randomization */ (7.11)

End for

End if

If fitness (q_e^{new}) < fitness ($q_{worst,e}$) /* Selection */

q''_e will replace $q_{worst,e}$ in next iteration;

Selection Rate^{++}

Else No change will occur in harmony memory

End if

Replacement Rate (RR) = Selection Rate^{++}/e (7.12)

/* Update Control Parameters */

$$HMCR_{e+1} = HMCR_{\min} + \frac{e}{e_{\max}} \bullet (HMCR_{\max} - HMCR_{\min}) \quad (7.13)$$

If RR > 0.2

$$PAR_{e+1} = PAR_{\min} + \tau \bullet (PAR_{\max} - PAR_{\min}); Where, \tau = \sqrt{\frac{1}{M}\sum_{m=1}^{M}(f_{v,e} - f_{v,e\text{-}1})^2} \quad (7.14)$$

For v=1: V

$$BW_{v,e} = BW_{\min} \bullet \frac{f_{e,\max}}{f_{e,\min}} \exp(bw_{v,e} \bullet e) \bullet RR \quad (7.15)$$

Where, $bw_{v,e} = \dfrac{\log\left(BW_{\max,v,e} \big/ BW_{\min}\right)}{e_{\max}}; BW_{\max,v,e} = \dfrac{Max_{v,e} - Min_{v,e}}{30}$

End for

Else

$$PAR_{e+1} = PAR_{\max} \text{ and } BW_{e+1} = BW_{\max} \quad (7.16)$$

End if

End for

7.4 Implementation of ATHS as Path Planning Algorithm:

When obstacle comes within specified range of on board sensors of underwater robot, proposed Adaptively Tuned Harmony Search algorithm will be activated as an obstacle avoidance strategy by following the flowchart of Figure 7.1. Generation of random population or Harmony memory has been done by following the mechanism described in Figure 5.5 of Section 5.4. All members in Harmony memory must be at safe distance from detected obstacle as given in eq. 5.14 of Section 5.4. Initialization step of ATHS algorithm has been completed by choosing the values of control parameters (as noted in Table 7.1) on trial and error basis.

Table 7.1: Parameters for Adaptively Tuned Harmony Search based navigational strategy

Parameters	Range of parameters
Harmony Memory Size (HMS)	30
Dimension of Memory	3
Harmony Consideration rate ($HMCR_e$)	$HMCR_{min}$ = 0.9, $HMCR_{max}$ = 0.99
Pitch Adjustment Rate (PAR_e)	PAR_{max} = 0.95, PAR_{min} = 0.1
Minimum value of Bandwidth (BW_{min})	0.0001 to 0.9
Maximum no. of iterations (e_{max})	200

Researchers [130] have suggested that low value of HMCR may worsen the optimization performance due to excessive randomness. HMCR near about 1 is more convenient to provide better performance. Therefore, $HMCR_e$ has been varied within the range as specified in Table 7.1 same as [140]. The HMS has found to be very sensitive towards the convergence speed of HS algorithm. Large HMS requires huge computational time for evolution and low HMS saves the memory space but may fail to provide better precision. HMS = 30 has found to be reasonable enough for three-dimensional path optimization problem. Initially, large value of PAR_e enhances the diversity population and linear reduction in value of PAR_e helps to achieve convergence at final stage. So, the range for PAR_e has been specified in Table 7.1. Choice of bandwidth parameter (BW_e) solely depends on the specifications of problem to be solved as found in previous applications of HS algorithm [138]. Variation of bandwidth parameter from high to low value has found to be preferable for present application to achieve a balance between diversification and intensification of search process. In pseudo code for ATHS, highest limit for BW_e (in eq. 7.15) has been defined to be adapted during navigation based on the boundaries of position variables which will be varied as robot progresses from one location to another. Minimum value for BW_e can be chosen from the range specified in Table 7.1 to provide safe three-dimensional navigation. It has been observed in eq. 7.8 and eq. 7.9 that sign of BW_e may affect the direction of search process, so, it has been chosen based on robot's position, as follows:

(a) For $\{rob_{x,k}, rob_{y,k}, rob_{z,k}\} < \{nob_x, nob_y, nob_z\} < \{goal_x, goal_y, goal_z\}$, positive value of BW_e will be preferred,

(b) When robot has to move in reverse direction as $\{goal_x, goal_y, goal_z\} < \{nob_x, nob_y, nob_z\} < \{rob_{x,k}, rob_{y,k}, rob_{z,k}\}$, then negative value of BW_e must be considered.

Before initiating the iterations of proposed version of HS, fitness of harmony memory has been evaluated by following eq. 7.6 of Pseudo code. Fitness function, effective for three-dimensional path planning algorithm, has already been designed in Section 5.4.1 of chapter Five. Flow chart of Figure 7.1 has shown that next global best position for underwater robot with respect to its current position can be found out by following the steps of proposed ATHS algorithm as narrated in Pseudo Code of Section 7.3.

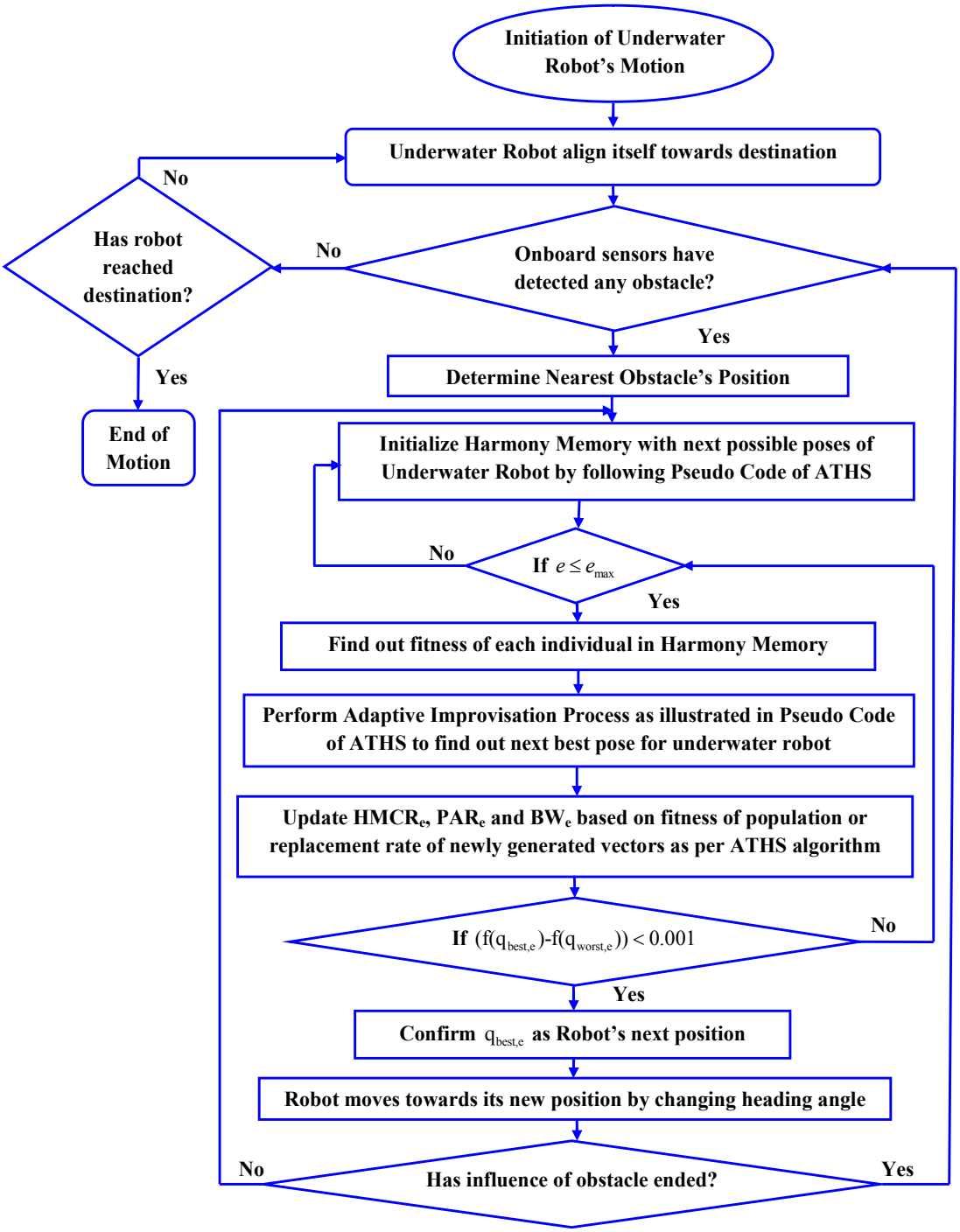


Figure 7.1 Flow chart for underwater navigation based on ATHS algorithm

Evolution process of ATHS has been allowed here to repeat for maximum 200 iterations (e_{max}) as given in Table 7.1. But before the e_{max} is completed, if the fitness difference between best fitted and worst fitted has found to be lower than specified threshold (0.001) value, then iteration will be stopped as given in flowchart (Figure 7.1).

7.5 Simulation Results and Performance Analysis

In this section, proposed adaptive version of HS algorithm has been employed as navigational strategy of underwater robot in MATLAB based simulated environments. Hypotheses required for carrying out the MATLAB based simulations have already been conferred in Section 4.5 of Chapter 4. A large number of simulations have been executed to verify navigational performance of proposed Adaptively Tuned Harmony Search algorithm.

7.5.1 Reactive Behaviors of Underwater Robot for ATHS based navigational strategy

Underwater must be reactive in nature with respect to the online sensory information while finding near optimal path between start and goal point of given scenario. Successful execution of preliminary robotic behaviours has been considered as most desirable performance for any navigational strategy. Therefore, Adaptively Tuned Harmony search algorithm has been employed as path planning algorithm in simulation scenarios (Figures 7.2-7.4) to perform robotic behaviours like Obstacle avoidance, Wall following and Target seeking etc.

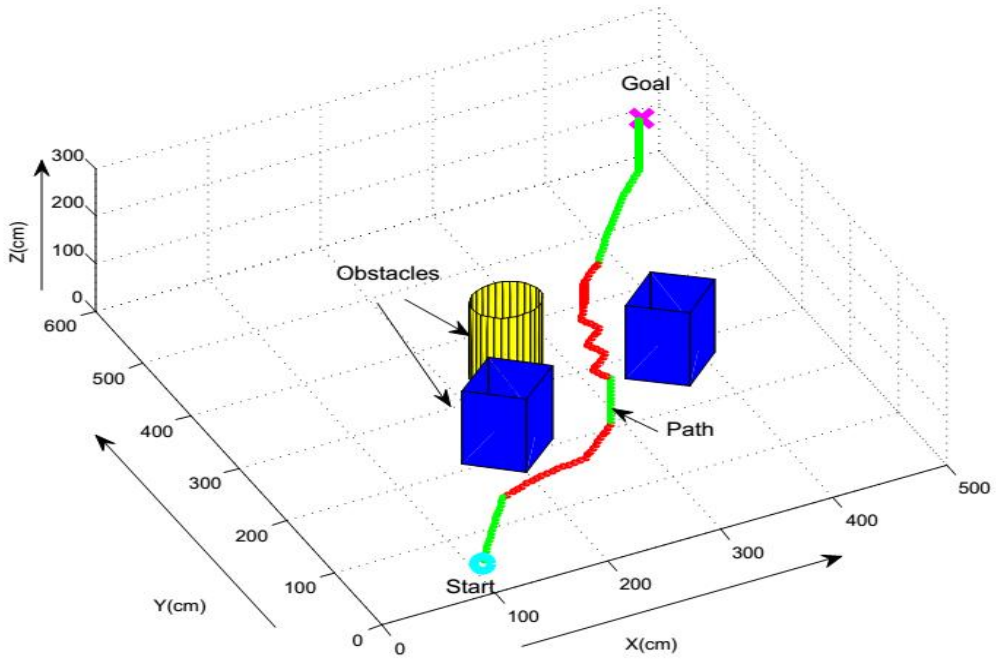


Figure 7.2: Three-dimensional navigation by proposed ATHS algorithm for small number of obstacles

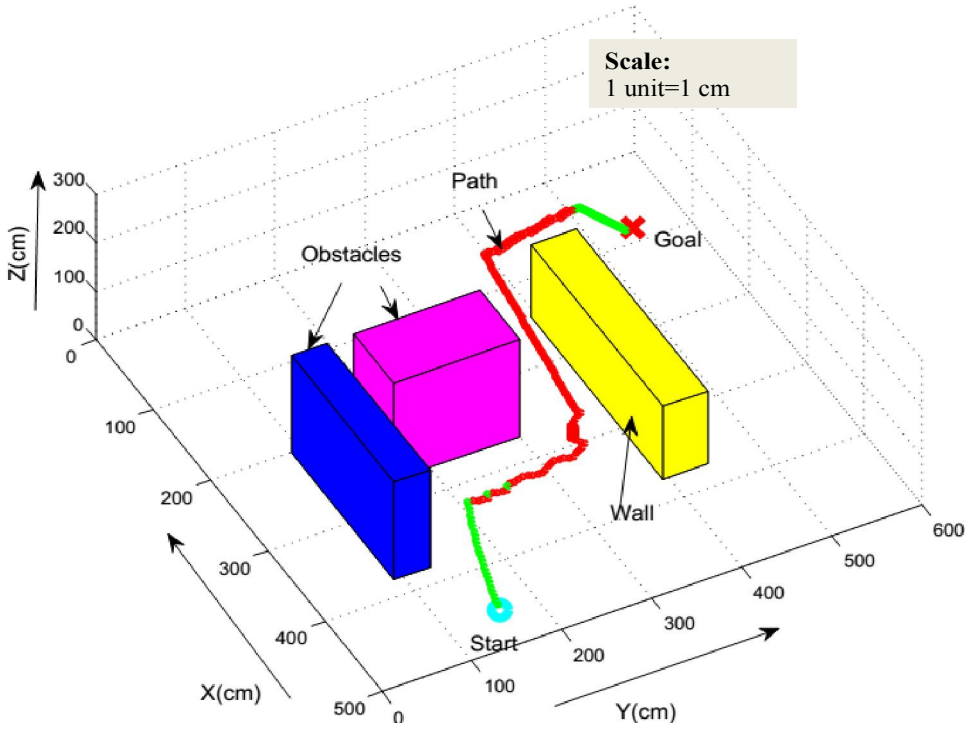


Figure 7.3: Wall following behaviour by proposed ATHS based Navigational Strategy

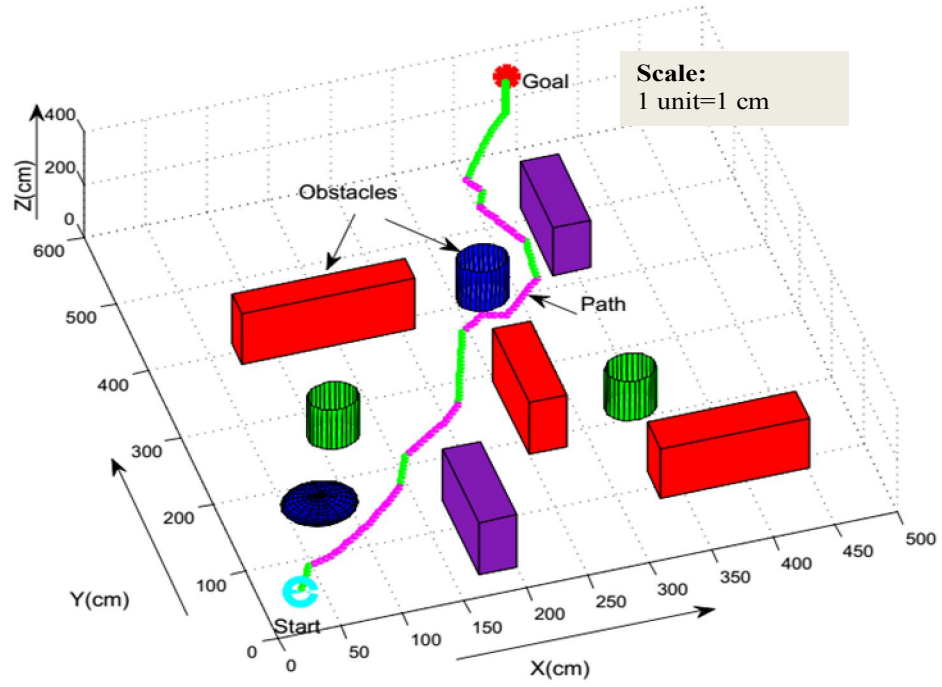


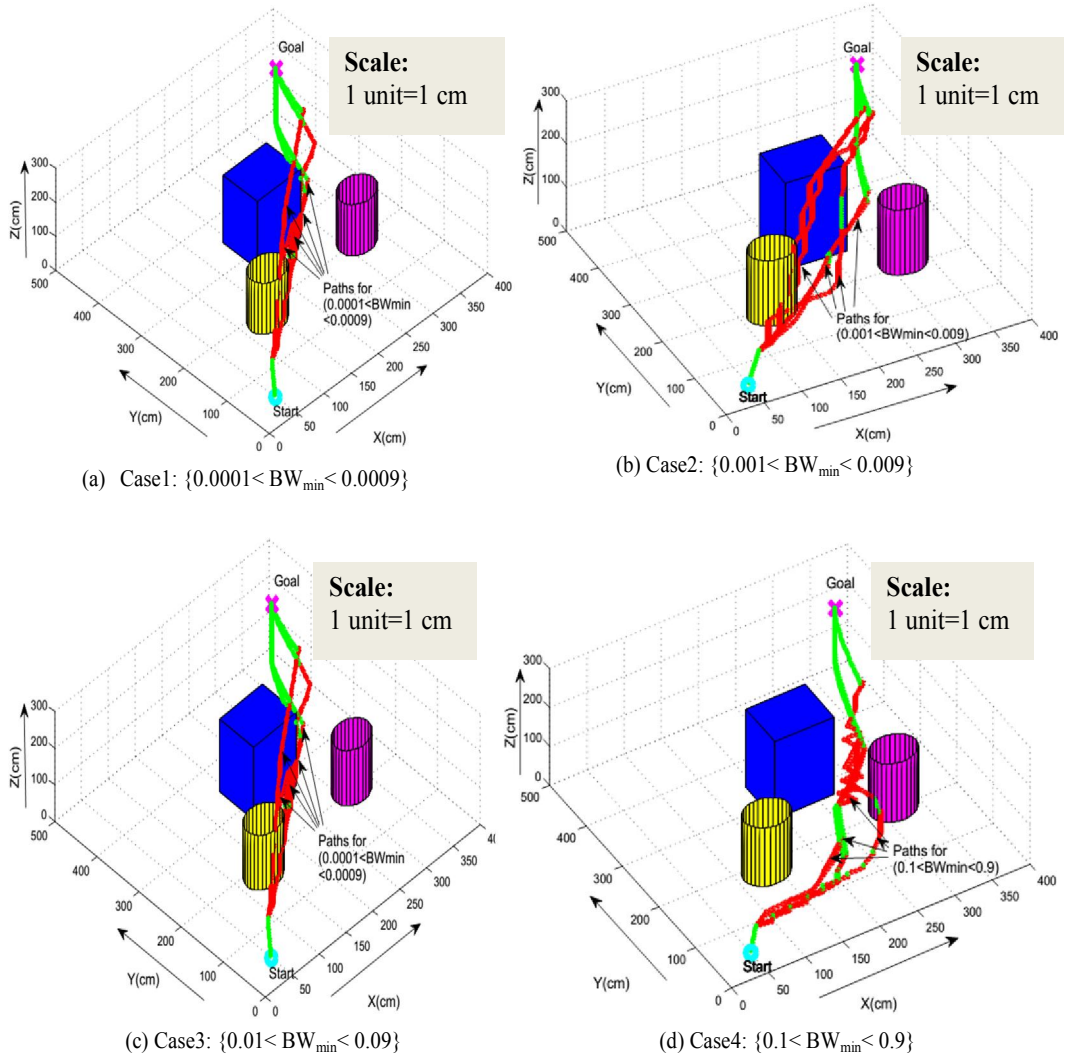
Figure 7.4: Obstacle Avoidance and Target Seeking behaviour of underwater robot for large number of obstacles

7.5.2 Selection of preferable BW_{min} value for Adaptively Tuned Harmony Search

In this section, a simulated analysis has been performed on trial and error basis to find out suitable value of BW_{min} which can aid safe clearance from obstacles during navigation. For this purpose, a navigational scenario (Figure 7.5) with three static obstacles has been considered in MATLAB based three-dimensional world. From extensive literature review it has been found that BW values can be chosen within the range from 0.0001 to 0.1 based on the attributes of optimization problems to be solved [138]. To improve the degree of accuracy while selecting value of BW_{min}, one more range of values (0.1 to 0.9) has been additionally considered here for three-dimensional navigation. Primarily, numerous simulations have been performed for the same three-dimensional scenario by changing BW_{min} within four ranges such as: {0.0001 to 0.0009} (case 1), {0.001 to 0.009} (case 2), {0.01 to 0.09} (case 3) and {0.1 to 0.9} (case 4) in Figure 7.5.

Table 7.2 List of path lengths as drawn by ATHS based navigational algorithm for different values of BW_{min}

Case Number	BW_{min} (Constant)	Path length (in cm)	Obstacle Avoided (Yes/No)
1 (Figure 7.5 (a))	0.0001	430.7	No
	0.0003	427.9	No
	0.0005	426.6	No
	0.0007	433.8	No
	0.0009	431.2	No
2 (Figure 7.5 (b))	0.001	459.4	Yes
	0.003	455.2	No
	0.005	457.5	Yes
	0007	451.4	No
	0.009	454.5	No
3 (Figure 7.5 (c))	0.01	437.9	No
	0.03	436.1	No
	0.05	448.2	Yes
	0.07	439.5	No
	0.09	457.3	Yes
4 (Figure 7.5 (d))	0.1	445.7	No
	0.3	**441.5**	**Yes**
	0.5	**443.2**	**Yes**
	0.7	449.8	No
	0.9	448.9	No



(a) Case1: {0.0001< BW_{min}< 0.0009}

(b) Case2: {0.001< BW_{min}< 0.009}

(c) Case3: {0.01< BW_{min}< 0.09}

(d) Case4: {0.1< BW_{min}< 0.9}

Figure 7.5: Three-dimensional simulated paths traced by proposed ATHS based navigational strategy for different values of BW_{min} (case1, case2, case3 and case 4)

By altering BW_{min} within each mentioned range, variation in path length and obstacle avoidance behaviour has been observed in Figure 7.5 and observations have been documented in Table 7.2. It has been found from Table 7.2 that desired navigational performance can be achieved only for BW_{min} values of 0.3 and 0.5. To find out more specific value of BW_{min} required for safe navigation and shorter path length, another set of simulations are performed by varying BW_{min} between 0.15 to 0.5 for same navigational scenario (Figure 7.6). Details of path length and obstacle avoidance behaviour for varying

BW_{min} values (0.15 to 0.5) have been illustrated in Table 7.3. For BW_{min} values in the range of 0.25 to 0.5, obstacles have been successfully avoided. When BW_{min} is 0.35, the shortest path has been traced by underwater robot in given scenario as depicted in Table 7.3. Therefore, BW_{min} has been kept fixed at 0.35 for proposed ATHS algorithm while applying as three-dimensional navigational strategy in any other scenarios.

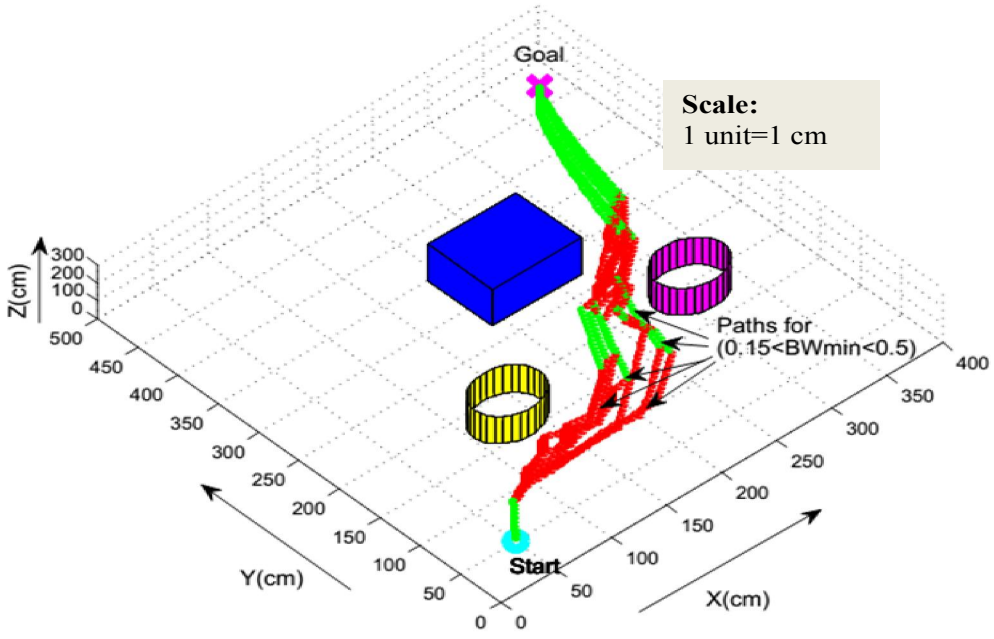


Figure 7.6: Three-dimensional simulated paths traced by using proposed ATHS algorithm by changing BW_{min} from 0.15 to 0.5

Table 7.3 Variation in path length for specific range of BW_{min} values

Sl. No.	BW_{min}	Path length (in cm)	Obstacle Avoided (Yes/No)
1	0.15	447.68	No
2	0.2	446.13	No
3	0.25	443.52	Yes
4	0.3	441.5	Yes
5	0.35	**439.24 (min.)**	Yes
6	0.4	442.57	Yes
7	0.45	441.94	Yes
8	0.5	443.27	Yes

7.5.3 Performance analysis of ATHS with respect to other variants of HS

In this section, proposed ATHS algorithm and other HS variants have been independently applied as navigational strategy for same simulation scenario (Figure 7.7) which contains three static obstacles of different shape and sizes between start and target point. Comparison among different versions of HS has been made regarding convergence speed and path length. Control parameters for respective algorithms have been specified separately in Table 7.4 by following their use in other applications. Population size (HMS=30), number of decision variables for each population member (V=3) and maximum permissible iterations (i_{max}=200) have been kept same for all algorithms to get a fair comparison. Each navigational strategy has to find out next best pose for robot in a sequential manner while avoiding obstacle close to robot's current pose in given scenario of Figure 7.7. Variation in fitness values for each algorithm has been recorded separately as an average of their 30 individual runs. Plotting of best fitness values with respect to iteration numbers can provide the average convergence performance for each algorithm as depicted in Figure 7.8.

Table 7.4: Details of control parameters for few variants of HS applied as path planning algorithm in given scenario (Figure 7.7)

Algorithms	Details of Control Parameters
GHS [131]:	HMCR=0.9, PAR_{min} = 0.01, PAR_{max}= 0.9.
SGHS [132]:	$HMCR_m$=0.98, PAR_m = 0.9, bw_{min}= 0.0005, $bw_{max} = \frac{UB - LB}{10}$, LP= 100
ITHS [137]:	HMCR=0.99, PAR_{min} = 0, PAR_{max}= 1.
IGHS [139]:	HMCR=0.99, PAR_{min} = 0.01, PAR_{max}= 0.99, bw_{min}= 0.0001, $bw_{max} = \frac{UB - LB}{20}$
Proposed ATHS:	Control parameters for pitch adjustment are used as per given details in Table 7.1.

Most popular variants of HS, such as, GHS [131], SGHS [132], ITHS [137] and IGHS [139] have been taken as reference to authenticate proposed optimization method as navigational algorithm. GHS, conceptualized by PSO, only considers best fitted solution as newly generated solution to be included in population. So, underwater robot driven by

GHS mechanism may get trapped in local minima situation. GHS [131] requires large time in average to reach a stable best fitness value compared to other HS variants as shown in Figure 7.8. SGHS [132], an improved version of GHS, demands large learning time to adapt its control parameters before finding out global best solution. So, average convergence speed of SGHS [132] in the context of present research area has also found to be slow enough in Figure 7.8.

ITHS[137] performs a number of stochastic searches within multiple subpopulations simultaneously to enhance search speed. Convergence speed of ITHS[137] for finding next best pose for robot has been found to be satisfactory up to a certain level. Moreover, fitness of best solution derived by ITHS [137] in current application is much lower than other HS variants except proposed ATHS algorithm (Figure 7.8). IGHS has already exhibited better performance than other HS variants (HIS [130], GHS [131], SAHS [134], and SGHS [139] etc.) while solving standard benchmark problems. In the present scenario, IGHS has shown desirable performance similar to proposed ATHS such as convergence speed remains slow initially but increases at final stage of iterations. But lowest fitness value computed by ATHS is less than IGHS as shown in Figure 7.8.

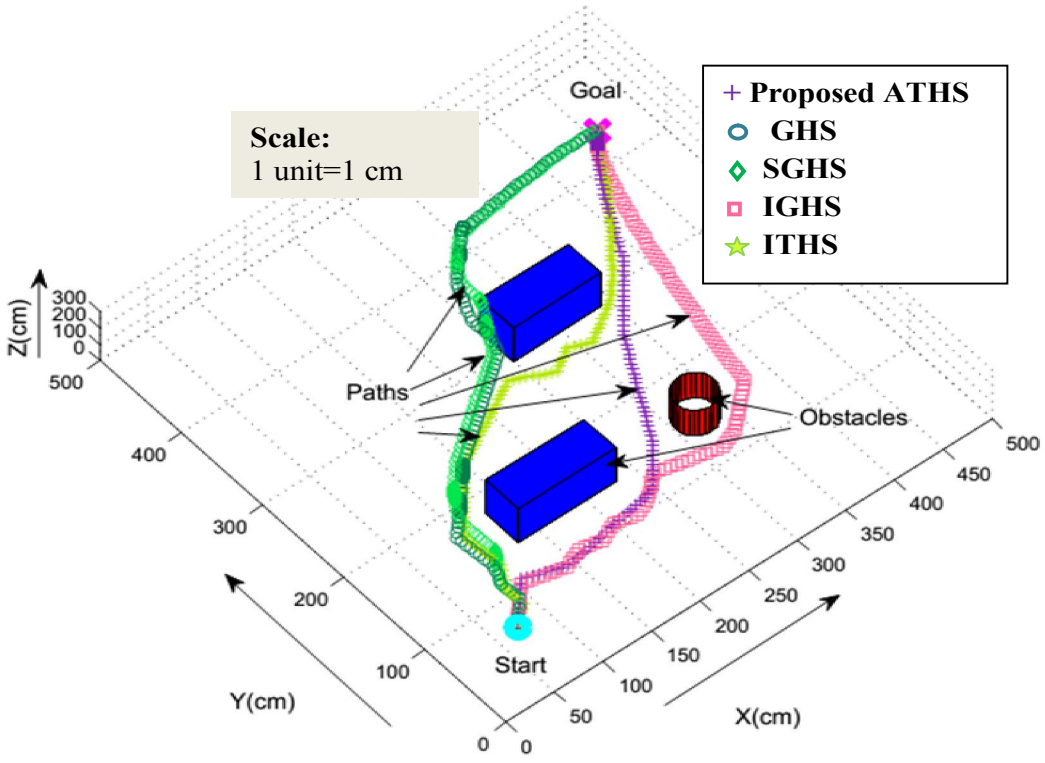


Figure 7.7: Simulated Paths traced by five different variants of HS algorithm

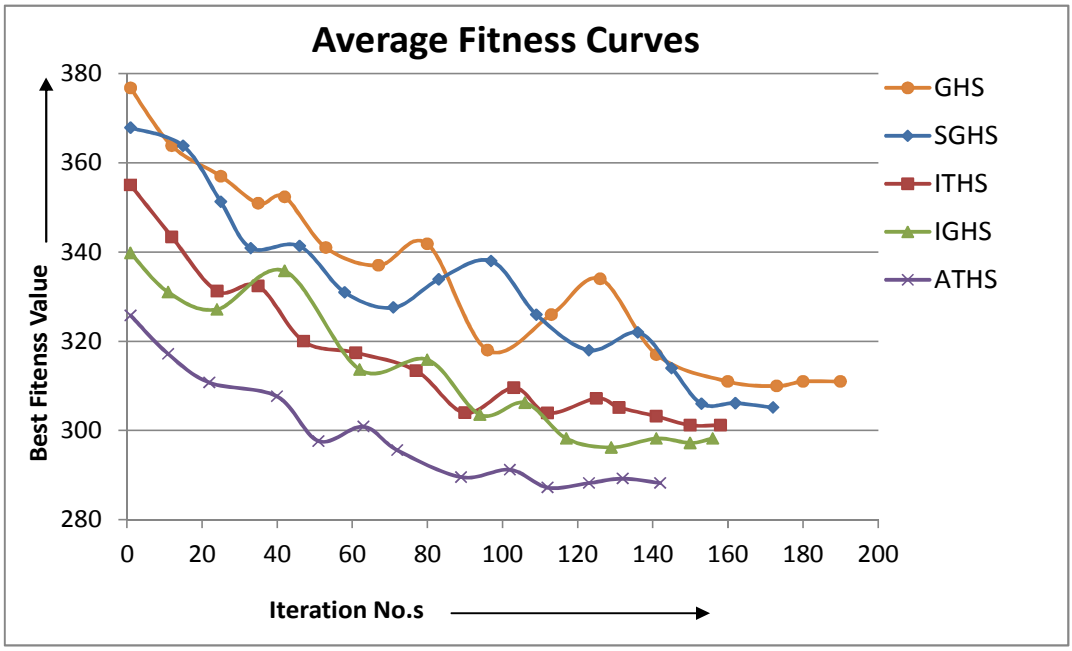


Figure 7.8: Average Fitness Curves for five HS variants while avoiding obstacle in given scenario of Figure 7.7

For every algorithm, the same simulation has been performed about 30 times and best path among them has shown in Figure 7.7. The diagonal distance between start and goal points (544.15 cm) has been considered as shortest path length of given scenario (Figure 7.7). Performances of above mentioned five algorithms have been evaluated regarding path length travelled by them, deviation in path length from optimum one, and also clearance from obstacle in Table 7.5. Possibilities of collision have been found for paths traced by GHS [131] and SGHS [132] in Figure 7.7. Path lengths for GHS [131] and SGHS [132] have been found almost same but larger than other HS variants in Table 7.5. Both IGHS [139] and ITHS [137] have traced paths which are safe from obstacles and path lengths have also been minimized compared to GHS [131] or SGHS [132].

Table 7.5: Comparison of proposed ATHS with few variants of HS with respect to Path Length and Obstacle Avoidance ability

Path Planning Algorithms	Path Length (in cm)	Deviation from Optimum path (in %)	Safe from collision (Yes/No)
GHS [131]:	603.89	9.89	No
SGHS [132]:	601.18	9.49	No
ITHS [137]:	581.37	6.40	Yes
IGHS [139]:	576.85	5.67	Yes
Proposed ATHS:	573.38	5.10	Yes

Proposed ATHS has exhibited more clearance from obstacles than IGHS [139] and ITHS [137] in Figure 7.7. Analyzing performance of above mentioned algorithms in Table 7.5, effectiveness of proposed adaptively tuned harmony search algorithm has been found during three-dimensional navigation with reduced path length and safe avoidance from obstacles.

7.5.4 Comparison with other heuristic approaches for three-dimensional path planning

In last few decades, numerous navigational methods have been introduced by researchers for underwater motion planning. Among conventional algorithms, potential field approach has gained sincere attention for providing solution to robot's path planning problem. Another promising evolutionary approach, Ant colony optimization has been preferred as motion planning strategy of underwater robot due to its simplicity, adaptability and robustness [103-104]. Therefore, HPF [25] and ACO [193] based underwater navigations have been chosen for performing comparative study with respect to proposed ATHS algorithm in simulation mode. Individual comparison has been illustrated as follows:

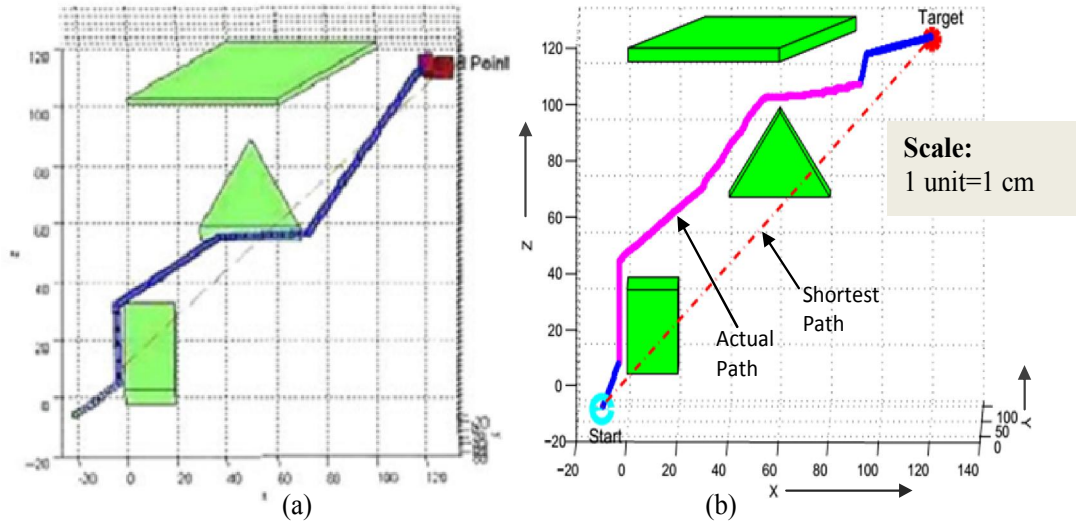


Figure 7.9: (a) Simulation result using Heuristic Potential Field as shown by Miao and Huang [25]; (b) View of Path obtained by implementing proposed ATHS approach.

(a) Heuristic potential field approach based navigation of AUV as introduced by Miao and Huang [25] has already been discussed in Section 6.5.3 of Chapter 6. Proposed Adaptively Tuned Harmony Search algorithm has been implemented as three-dimensional path planner in simulation scenario of Figure 7.9(b) which is similar with Figure 7.9(a). Path travelled by proposed navigational scheme has found to be safer than the path shown in Figure 7.9(a) during obstacle avoidance. The comparison between simulation results regarding path length has been portrayed in Table 7.6. Marginally shorter path length has been traced by recently proposed metaheuristics based strategy than HPF [25].

(b) In spatial space, Guanglei and Heming [193] have employed improved Ant colony optimization as AUV's path planning based on Octree model of search space. Potential field's concept of attraction and repulsion has been employed to develop heuristic information which has been found to be beneficial for improved version of ant colony optimization. Performance of Improved ACO for avoiding obstacles has been exhibited in Figure 7.10(a) with dimension of (850m X 450m X 250m). It has been claimed by Guanglei and Heming [193] that improved version of ACO requires less computation time than grid based basic ACO algorithm during three-dimensional navigation for same simulation environment [193].

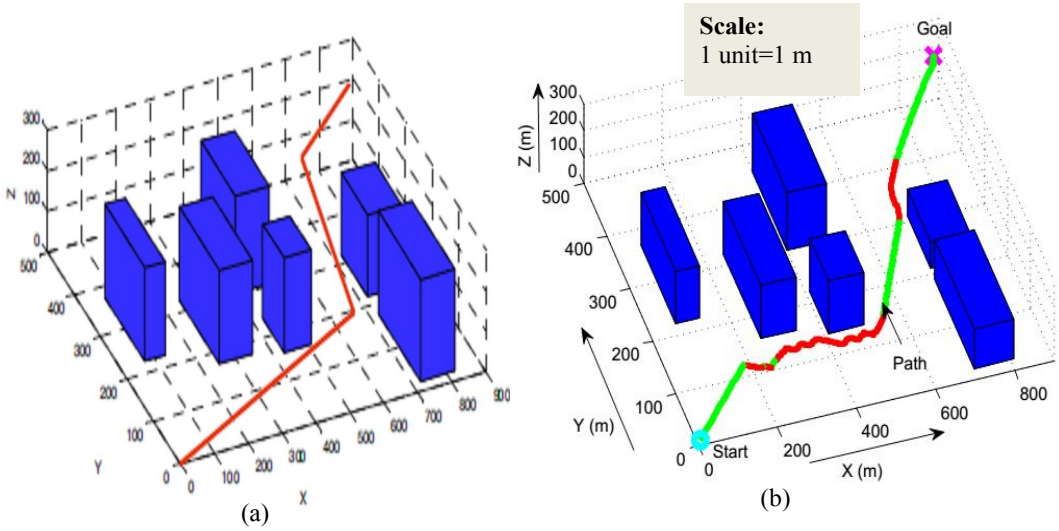


Figure 7.10: (a) 3-D Simulation result using Improved Ant Colony Optimization as shown by Guanglei and Heming [193]; (b) View of Path obtained by implementing proposed ATHS based approach in simulated environment same as Figure 7.10(a)

Three-dimensional navigation based on proposed ATHS algorithm has been executed within simulation scenario of Figure 7.10(b) where obstacle arrangement has been created same as Figure 7.10(a). The simulated path of Figure 7.10(b) has found to be free from sharp edges and possesses safe clearance from obstacles. Comparison between navigational performances of proposed ATHS and improved ACO algorithm [193] has shown minor variance in path length (Table 7.6). Good agreement during comparison may authenticate the feasibility of proposed Adaptively Tuned Harmony Search algorithm as navigational strategy of underwater robot.

Table 7.6 Comparison between Proposed ATHS algorithm and other navigational strategies with respect to simulated path length

Figure No.	Navigational Strategy	Length of 3D path traced by underwater robot (in cm)	Deviation (in %)
7.9(a)	Heuristic potential field approach [25]	252.85	2.48
7.9(b)	Proposed Adaptively Tuned Harmony Search approach	246.73	
7.10(a)	Improved ACO [193]	1087.85	1.03
7.10(b)	Proposed Adaptively Tuned Harmony Search approach	1076.73	

7.6 Experimental Results and Comparisons

Navigational strategy based on proposed new version of HS has been incorporated within the controller of underwater robot "GNOM-Baby" to exhibit its robotic behaviors in real world underwater environment. In experimental mode, underwater robot GNOM-baby has started its motion from stage 1 of Figure 7.11 which has shown scattered positions of obstacles in between specified start and goal points of robot. Experimental scenario of Figure 7.11 has been arranged by following the simulation scenario of Figure 7.4. Initially, motion of underwater robot has been found to be oriented towards the goal position. During obstacle avoidance, the direction of motion has been changed to follow the best fitted locations as estimated by Adaptively Tuned Harmony Search method. Stage 2 of Figure 7.11 can be considered as one position of underwater robot while avoiding nearest obstacle. As underwater robot has progressed towards its goal, it has faced different critical situations while avoiding obstacles. Stages 3 to 5 of Figure 7.11 have shown different locations of underwater robot while following near optimal path within real time underwater environment.

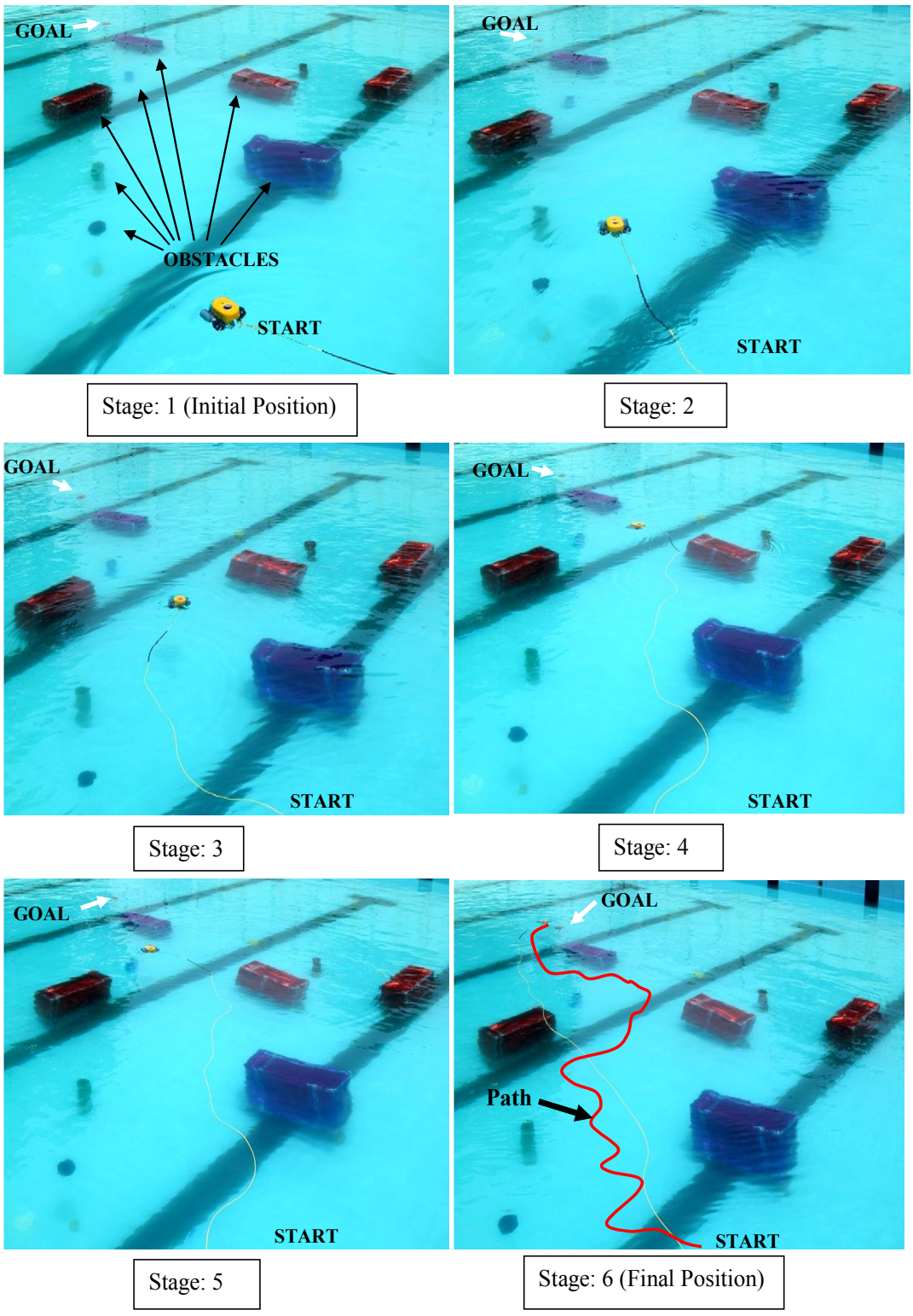


Figure 7.11: Experimental view for proposed ATHS based underwater navigation of GNOM Baby for obstacle arrangement near about same as in Figure 7.4

Stage 6 of Figure 7.11 has shown that underwater robot has successfully reached the specified goal point of given scenario. Both obstacle avoidance and goal seeking behaviors of underwater robot have been effectively executed during navigation in experimental mode.

7. 6.1 Comparison between simulation and experimental results:

To get rid of any uncertainty hidden within heuristic approach based navigational strategy, navigation of underwater robot has been repeatedly performed for 20 times within each scenario of simulation and experimental modes. Authenticity of proposed novel navigational strategy has been explored by comparing simulated and experimental performance of robot for same scenario (as seen in Figure 7.4 and 7.11 respectively).

Table 7.7 Comparison between experimental and simulated results for path length in Scenario1 (Figure 7.4 and 7.11)

1	2	3	4	5	6
No. of Runs	**Path length in simulated mode (cm)**	**Deviation from Optimum path (%)**	**Path length in experimental mode (cm)**	**Deviation from Optimum path (%)**	**Difference between column 2 and 4 (%)**
1st	729.7	6.27	763.1	4.94	4.58
2nd	718.8	4.68	761.8	4.77	5.99
3rd	723.8	5.42	765.7	5.30	5.78
4th	720.7	4.95	762.7	4.88	5.83
5th	727.2	5.90	767.2	5.51	5.51
6th	719.6	4.80	766.6	5.42	6.53
7th	713.8	**3.96**	760.2	4.54	6.49
8th	721.5	5.08	764.1	5.07	5.88
9th	720.3	4.90	762.2	4.83	5.82
10th	724.5	5.52	764.7	5.07	5.45
11th	719.2	4.75	756.4	4.02	5.16
12th	716.9	4.40	765.9	5.33	6.83
13th	723.5	5.37	761.5	4.73	5.25
14th	724.9	5.57	759.1	4.39	4.71
15th	731.2	6.48	763.2	4.95	4.38
16th	722.2	5.18	758.5	**4.32**	5.03
17th	725.7	5.69	760.4	4.58	4.78
18th	728.3	6.06	766.3	5.38	5.22
19th	724.8	5.55	764.7	5.16	5.50
20th	715.9	4.26	763.9	5.05	6.70
Average difference in path length between simulated and experimental results					5.57

Table 7.8 Comparison between experimental and simulated results for travel time in Scenario1 (Figure 7.4 and 7.11)

1	2	3	4
No. of Runs	**Travel time in simulated mode (in sec)**	**Travel time in experimental mode (in sec)**	**Difference between column 2 and 4 (in %)**
1st	26.06	27.55	5.71
2nd	17.11	18.01	5.24
3rd	21.94	23.13	5.46
4th	19.48	20.56	5.54
5th	25.08	26.61	6.13
6th	18.45	19.44	5.37
7th	15.86	16.71	5.32
8th	20.04	21.16	5.59
9th	19.18	20.22	5.43
10th	22.64	23.94	5.75
11th	18.49	19.46	5.27
12th	16.76	17.75	5.94
13th	21.28	22.38	5.16
14th	22.65	23.70	4.62
15th	27.08	28.40	4.88
16th	21.12	22.16	4.91
17th	23.41	24.52	4.75
18th	25.31	26.86	6.14
19th	22.77	23.95	5.17
20th	15.91	16.93	6.40
Average difference in travel time between simulated and experimental results			5.44

Path length and travel time for each run of navigation based on Adaptively Tuned Harmony Search in simulated and experimental modes have been noted in Table 7.7 and 7.8 respectively. Actual path length has been compared with the shortest distance between start and goal positions which has been recorded as 686.7 cm and 727.2 cm for simulation (Figure 7.4) and experimental (Figure 7.11) scenarios respectively.

Moreover, comparisons between a number of simulation and experimental scenarios with different environmental conditions have been performed to ensure the accuracy of proposed navigational strategy. Observations from comparative study have been listed in Table 7.9 and 7.10 for path length and travel time respectively.

Table 7.9 Comparisons between experimental and simulated results for five more scenarios regarding path length

Different Scenarios	Best path length in Simulation (in cm)	Best path length during Experiment (in cm)	Difference (in %)
Scenario-2	791.5	835.2	5.49
Scenario-3	535.7	563.4	5.17
Scenario-4	674.5	711.4	5.48
Scenario-5	846.7	887.1	4.77
Scenario-6	939.6	993.2	5.69

Collision-free path traced by underwater robot with minimum deviation from shortest path length has been considered as best one. Minor differences in average have been recorded between simulated and experimental performances of recent navigational strategy regarding path length and travel time.

Table 7.10 Comparisons between experimental and simulated results for five more scenarios regarding travel time

Different Scenarios	Travel time during simulation (in sec)	Travel time during experiment (in sec)	Difference (in %)
Scenario-2	21.39	22.47	5.04
Scenario-3	10.93	11.50	5.15
Scenario-4	18.23	19.23	5.49
Scenario-5	29.20	30.78	5.41
Scenario-6	32.40	34.13	5.32

7.7 Summary

In this chapter, exactness and randomness of search process have been tried to balance by introducing exploitative nature within the global optimization process of Harmony Search algorithm. Several attributes of proposed Adaptively Tuned HS and its performance during underwater navigation has been summarized as follows:

- Adaptive tuning of control parameters based on fitness of recent population and automatic selection of perturbation schemes for solution vectors in pitch adjustment process have been introduced here to balance intensification and diversification abilities of HS algorithm. Proposed adaptive version of HS has been employed as navigational strategy of underwater robot.

- At initial phase of obstacle avoidance, global search based on HS principle may properly explore the search space which contains next possible poses of robot. As iteration progresses, newly incorporated adaptive local search approach has been executed to locate next best pose for underwater robot which is very close to optimal position. Thus proposed Adaptively Tuned Harmony Search (ATHS) algorithm can maintain explorative and exploitative nature of search during three-dimensional path finding process.

- The proposed HS algorithm has been compared with other variants of HS for three-dimensional navigation in simulated workspace. Analysis on comparison has portrayed the benefits of ATHS algorithm over other HS variants regarding convergence speed, fitness value, path length and obstacle avoidance behaviour during navigation.
- Navigational performance of proposed algorithm has been found to be better than other path optimization methods (Potential field method and Ant colony optimization) for generating near optimal path in simulated three-dimensional environment.

- While executing essential robotic behaviors, navigational performance of proposed ATHS method have been verified through numerous simulated and experimental investigations in a realistic manner.

- Collision avoidance, shorter path length and minimum computational time have been considered as key features for evaluating simulation and experimental results in different scenarios. Comparison between simulation and experimental results has shown average error of 5.36% for path length and 5.31% for travel time. Insignificant difference between simulation and experimental results has proven the robustness of proposed navigational strategy.

Analysis on recent investigation has certified that proposed Adaptively Tuned Harmony Search (ATHS) algorithm is capable of tracing collision free near optimal three-dimensional path for underwater robot. Present chapter has mainly focused on tuning mechanism of control parameters for improving local search ability of harmony search process. In next chapter, some deterministic nature has been tried to be assimilated within HS by hybridizing it with other stochastic approach.

8. Hybridization of DE and HS Approaches for Navigation of Underwater Robot

Adaptation of control parameters has been considered as an amicable way to improve navigational performance of EA based three-dimensional path planning algorithm in previous chapters. Performed investigations have observed that reasonably near optimal path has been traced by underwater robot embedded with proposed adaptive version of individual heuristic approach. To increase the degree of optimality in stochastic optimization based path planning algorithm, current chapter has proposed hybridization of two promising evolutionary approaches (DE and HS) which requires less mathematical complexity during evolution.

8.1 Introduction

Autonomous motion within unpredictable underwater scenario inevitably requires accurate determination of waypoints by the navigational controller of underwater robot. Different aspects of evolutionary based navigational strategies for underwater robot have been analyzed in present dissertation. Due to stochastic nature, any standard heuristic search may provide suboptimal solution rather than global one. Extensive literature review of Chapter 2 has indicated that optimization performance of regular EAs can be improved by adjusting their control parameters or by combining stochastic and deterministic behaviors of two individual metaheuristics. Adaptive versions of heuristic approaches (e.g. SFLA, DE and HS) have already shown quite feasible performances for minimizing path length and avoiding obstacles safely during underwater navigation. But discrete application of evolutionary approaches may face intricacies related to large memory requirement, long computation times or premature convergence etc. [147]. Hybridization of evolutionary approaches may effectively overcome such drawbacks while solving complex nonlinear optimization problems of large dimension [88].

Basically, the search quality of hybridized optimization process has been upgraded by mingling strong attributes of different optimization algorithms. For example, DE requires less computational time to provide better approximation of solutions comparative to other EAs. On the other hand, HS can be recognized as an efficient global optimization mean [192]. Therefore, combination of DE and HS algorithms can successfully balance the explorative and exploitative behaviours of search process by employing their respective

benefits [194]. Faster convergence speed and local minima avoidance ability of proposed hybridization of DE and HS algorithms has inspired present research work for implementing it as three-dimensional path optimization method during underwater navigation. Practicability of proposed DE-HS method may be confirmed by comparing it with existing versions of HS and DE in terms of convergence speed. Moreover, Simulation and experimental analysis on hybrid DE-HS based path planning strategy may reveal the optimization ability of proposed hybridization process.

8.2 Benefits of Proposed DE-HS Hybridization

HS's ability to deal with imprecision, vagueness and high-dimensionality makes it popular to solve many real world complex problems for last few years. HS's new vector generation strategy enriches the diversity of population which enables HS to avoid premature convergence. Flexibility, simplicity and low probability of being trapped in local minima facilitate HS to act as global optimization scheme for path planning problems [141-145]. In pitch adjustment process of conventional HS, vectors are perturbed by randomly chosen bandwidth value which may aid the exploration ability but may not provide faster convergence speed like DE. Therefore, conventional HS of stochastic nature may fail to guide the newly generated vector towards its optimum value [188].

On the other hand, Storn and Price [178] have achieved fast convergence speed through fine perturbation of solution vectors in Differential Evolution scheme which is a simplified version of Genetic Algorithm. DE requires few control parameters to generate new vector through simple arithmetic operation. Easy calibration and stochastic nature of DE has found to be very effective as local path planning scheme for the robot to avoid obstacles [121-125]. However, a new solution formed using a set of few randomly selected individuals may limit the exploration ability in DE when the population diversity is low [195]. Improper values for control parameters and wild exploitation nature of DE may lead towards early convergence.

HS and DE can mutually employ the advantages of each other's strength to overcome their respective drawbacks [191, 196]. Coupling between HS and DE may follow two separate ways: HS's random search mechanism can be applied on the population evolved by DE process to explore the search space in a wide manner. In other way, DE's fine tuning ability may also be inserted within pitch adjustment step to enhance the

convergence speed [194]. In present research work, adaptive versions of both mechanisms have been employed in a cooperative manner to trace collision-free optimal path for underwater robot. Initially, wide exploration of search space may be achieved by following global search scheme of Harmony search. Such mechanism may assist to identify the region around best fitted solution. As iteration progresses, deterministic nature of DE's mutation strategy will help to fine tune the solutions towards optima. Convergence speed of DE and search quality of HS are the desirable benefits of proposed hybridized optimization algorithm which has been employed here to find out the next best pose for robot on detection of obstacles by on board sensors during navigation.

8.3 Hybridization of Differential Evolution and Harmony Search Algorithms

In hybridization process, mutation strategies of DE have been borrowed to perform the pitch adjustment operation of HS. Both memory consideration and pitch adjustment rules of conventional HS have focused on strengthening the exploration ability of search process. Execution of crossover operation after pitch adjustment process embedded with differential mutation schemes can enhance the exploitation ability in proposed hybrid DE-HS algorithm. Therefore, new vector generation strategy of DE-HS approach may access each dimension of a solution vector for fine-tuning. In proposed DE-HS algorithm, mutation step size and execution probabilities of DE-HS (HMCR, PAR, and CR) have been varied in an online manner based on fitness value of population and current iteration number. Replacing the concept of bandwidth parameter (BW_i), scaling factor (F_i) has been employed to scale the mutation step size same as DE algorithm. These features may increase the flexibility of search process to get better solutions along with faster convergence speed. Adaptation of DE's mechanism at different stages of Harmony Search (Figure 8.1) has been illustrated stepwise as given below:

Steps of Hybrid DE-HS Method:

Step 1: *Initialization*

In proposed hybrid DE-HS algorithm, initialization of solution vectors in harmony memory and their fitness evaluations have been performed by following eq. 7.5 and eq. 7.6 of Pseudo code for ATHS algorithm as described in previous chapter. Control parameters of both DE and HS have also been initialized within their respective

boundaries same as proposed Dynamic DE and Adaptively Tuned Harmony Search algorithms respectively.

Step 2: *Harmony Improvisation Process Improved by DE*

Among three rules of conventional Harmony Search process, only pitch adjustment rule has been reformed by incorporating differential mutation schemes for perturbing solution vectors in a controlled way. Rules for HMCR and randomization have been kept same as original HS. After completion of pitch adjustment process, perturbed solution vector has been evolved one more time by following crossover mechanism of DE to generate new solution vector. Consideration of each dimension of solution vector in crossover may add more diversity in solutions of proposed DE-HS algorithm. Pseudo code for proposed DE based Harmony Improvisation Scheme has been formulated and described as follows:

Step2a: Adaptive pitch adjustment process based on DE mechanism

To enhance local search ability of optimization process, three differential mutation strategies: DE/best/2, DE/current to best/2 and DE/rand/2 have been considered in a collaborative manner to perturb the solution vector (v_p) chosen by $HMCR_i$ probability. Any one of them (as given in eq. 8.2, 8.3 and 8.4) will be activated in one iteration based on probabilistic value of PAR_i and number of iterations as specified in Pseudo code. Solution vectors participated in mutation schemes denoted by v_q, v_r, v_s and v_t are chosen randomly from harmony memory. Control parameter, F_i has been used to scale the difference vectors in mutation strategy. To avoid premature convergence, variation of mutation schemes will be depended on stages of the evolution.

At initial phase of evolutions $i \ll i_{max}$, PAR_i will be kept as high. So, probability of using differential mutation scheme DE/rand/2 (eq. 8.4) will be high to provide diversity in population. As iteration progresses, PAR_i will decrease and probability of choosing differential mutation schemes given in eq. 8.2 or 8.3 will be high. In the middle phase of evolutions $i < i_{max}$, activation probability of DE/current to best/1 (eq. 8.3) will be high balancing both exploration and exploitation as the considered mutation strategy is a combination of random difference and difference of current vector with best fitted vector. As iteration increases more, $i \approx i_{max}$, DE/best/2 (eq.8.2) will be mostly chosen for providing faster convergence by directing all vectors towards best fitness solution. Two-

pairs of differences in mutation strategy may yield more accurate perturbation than one-paired mutation scheme because of its Gaussian behavior [8, 39].

Step2b: DE's Crossover mechanism

Based on a comparison between a random number and crossover probability (CR_i), a trial vector v_i^{new} has been regenerated by taking the values of parameters from either newly generated vector v_i'' (eq. 8.5) or chosen population member v_i' (eq. 8.6). Only binomial type crossover has been implemented here to reinforce the exploitation ability.

Pseudo-code for DE based Harmony Improvisation Process:

For i=1: i_{max} /* Iteration started*/

If $U(0,1) \le HMCR_i$ /* Improved Improvisation Process */

$v_i' = v_{p,i}$ /* p^{th} solution vector has been chosen from Harmony Memory */ (8.1)

For d=1: D /* For each dimension of chosen solution vector */

If $U(0,1) \le PAR_i$ /* DE based Pitch Adjustment Process */

If $U(0,1) \ge \frac{i_{max} - i}{i_{max}}$

$v''_{d,i} = v_{d,best,i} + rand(0,1) \cdot F_i \cdot \{(v_{d,p,i} - v_{d,q,i}) + (v_{d,r,i} - v_{d,s,i})\}$ /* DE/best/2 */ (8.2)

Else

$v''_{d,i} = v'_{d,i} + rand(0,1) \cdot F_i \cdot \{(v_{d,best,i} - v_{d,p,i}) + (v_{d,q,i} - v_{d,s,i})\}$ /*DE/current-to-best/2*/ (8.3)

End if

Else $v''_{d,i} = v'_{d,i} + rand(0,1) \cdot F_i \cdot \{(v_{d,q,i} - v_{d,s,i}) + (v_{d,r,i} - v_{d,t,i})\}$ /* DE/rand/2*/ (8.4)

End if

End for

For d=1: D

If $(rand(0,1) \le CR_i)$ /* Recombination like DE */

$v_{d,i}^{new} = v''_{d,i}$ (8.5)

Else $v_{d,i}^{new} = v'_{d,i}$ (8.6)

End if

End for

Else

For d=1: D

$$v_{d,i}^{new} = v_{d,i}^{\min} + rand(0,1)\bullet(v_{d,i}^{\max} - v_{d,i}^{\min}) \qquad \text{/* Randomization */} \tag{8.7}$$

End for
End if
End for

Step2c: Randomization

Solution vectors with (1-HMCR) probability will be generated from outside harmony memory but within specified range of search space by following eq. 8.7 and will not be participated in crossover operation.

Step3: *Selection*

Newly generated vector with better fitness than worst fitted solution vector of present population will replace the worst one in next iteration. In current iteration, if the newly generated vector is fitted enough to be accepted in harmony memory then selection rate will be increased once.

Step4: *Update Control Parameters of HS and DE in an Adaptive Manner*

Adaptation of control parameters (F_i and CR_i) of DE based optimization has already been described in eq. 6.19 to eq. 6.21 of Chapter 6 which has been followed here without any alteration. Similarly, control parameters of HS ($HMCR_i$ and PAR_i) have also been updated by following fitness based adaptation rules as proposed in previous chapter.

Steps 2, 3 and 4 will be reiterated until the termination criterion (maximum iteration or negligible difference in fitness between best fitted and worst fitted solutions of current population) has been satisfied as shown in Figure 8.1.

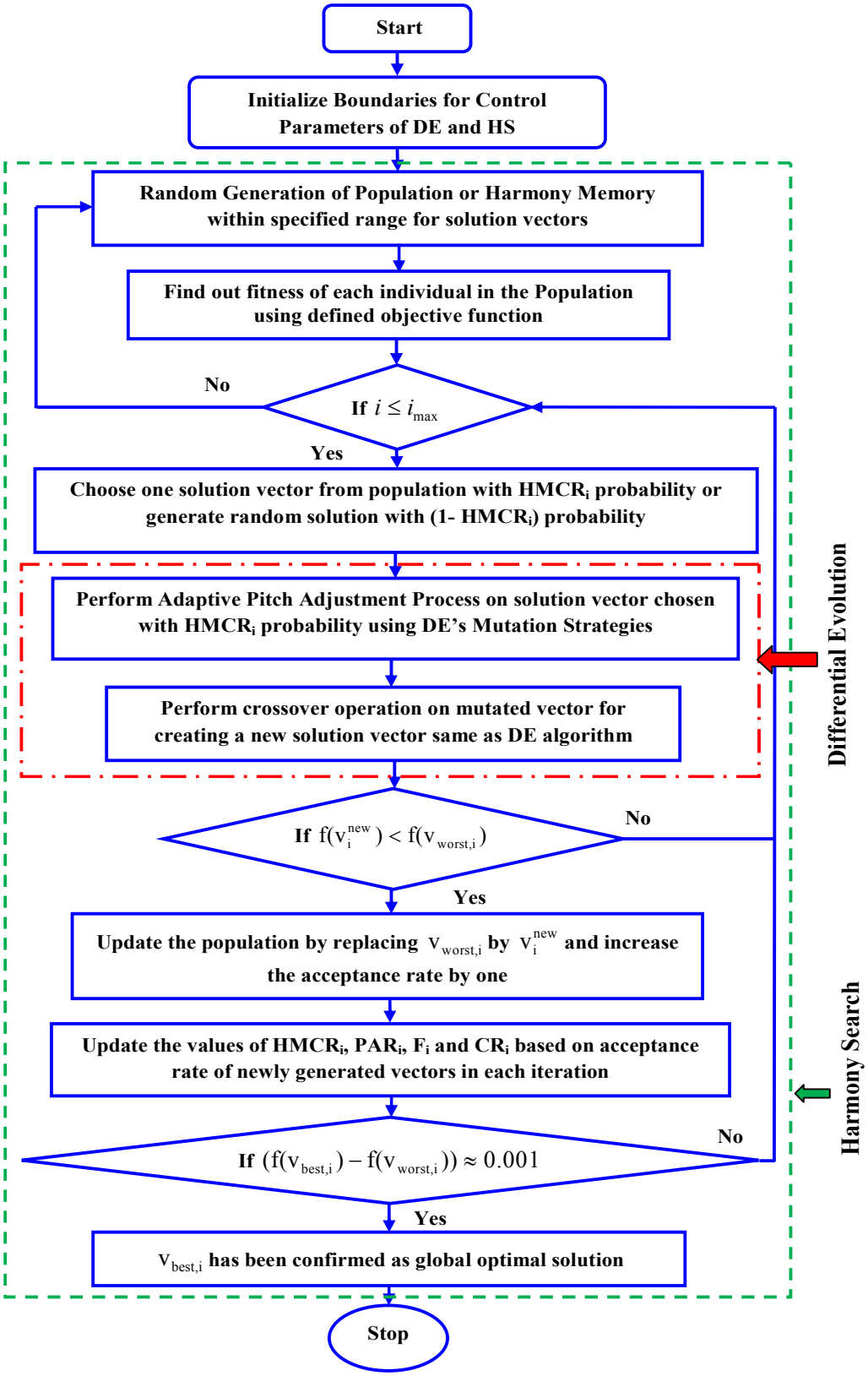


Figure 8.1 Flow chart of Hybrid DE-HS algorithm

8.4 Implementation of Hybrid DE-HS as Path Planning Algorithm:

Realization of Hybrid DE-HS algorithm as navigational strategy of underwater robot has followed the same process like DDE or ATHS algorithm which has already been described in previous two chapters. Control parameters for DE and HS processes have been chosen by following Table 6.1 and 7.1 respectively. Same as other proposed metaheuristics, hybrid DE-HS has been applied as obstacle avoidance strategy during underwater navigation. For avoiding obstacles, proposed hybrid DE-HS algorithm has to optimize next global best position with lowest fitness value from the randomly generated population of underwater robot's probable positions. Optimization process has been executed by following the steps as given in flowchart of Hybrid DE-HS algorithm (Figure 8.1). Apart from safe clearance from obstacles, path length minimization has been considered as an optimization criterion in the fitness function for three-dimensional path planning algorithm which has already been designed in Section 5.4.1 of Chapter 5. On completion of iterations of hybrid DE-HS algorithm, underwater robot will change its heading angle to reach the estimated global optimal position. Determination of next position of robot based on Hybrid DE-HS algorithm will be recurred until the effect of obstacles come to an end. In obstacle-free region, motion of underwater robot will be oriented towards the goal of given scenario.

8.5 Simulation Results for Hybrid DE-HS based Path Planning Algorithm

Before performing any real time application, three-dimensional path planning based on hybrid DE-HS algorithm has been executed in a large number of simulated environments. By following few common assumptions on MATLAB based simulations (as given in Section 4.5 of Chapter 4), proposed navigational strategy has been implemented to avoid obstacles while moving towards target of given scenario. The computational time required for searching next best position of underwater robot completely depends on convergence speed of proposed hybrid DE-HS algorithm.

8.5.1 Performance analysis of Hybrid DE-HS with respect to other metaheuristics

A simple simulation environment (500x500x500 cm) has been considered in Figure 8.2 to evaluate performance of proposed Hybrid DE-HS algorithm in comparison with other eleven metaheuristics. In the given scenario, underwater robot has started at point S

(150,10,0) and stimulated to reach target at point E (230,470,230). Between S and E, there are two rectangular shape obstacles whose centre points have been located at O1 (180,160,100) and O2 (295,310,100) respectively. On board sensors of underwater robot may sense the presence of 1st obstacle at point A (125,130,45) which is at a safe distance from the boundary defined for obstacle 1.

Now optimization algorithm has been actuated to find out next best possible pose for robot in current scenario so that robot may not collide with the obstacle. Robot will move towards that derived pose. This process of finding next best position will be repeated up to point B (250,200,110) where the influence of first obstacle will come to an end. Next, robot will follow the target angle to progress towards target and but again it will sense 2nd obstacle at point C (245,280, 150). The previous process will be repeated to avoid collision successfully. Finally at point D (200,340,190), underwater robot may found itself out of obstacle enriched area and directly headed towards goal at point E. Change in colour symbolizes activation of obstacle avoidance (red) or target seeking (green) behaviours of robot.

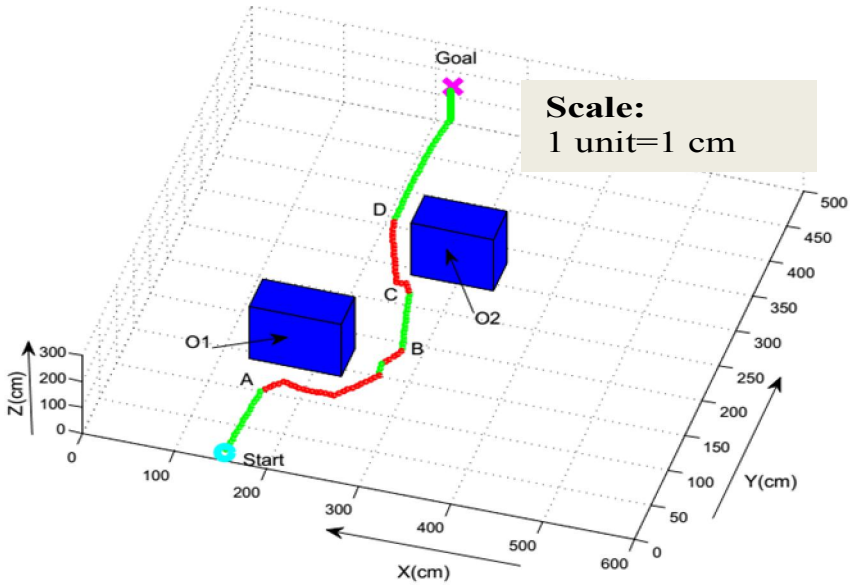


Figure 8.2 Simple Simulation Environment with Two Obstacles only

Due to presence of obstacles, actual path length during navigation will be more than optimum one (534.63 cm) which is a straight distance between start and goal point in given scenario (Figure 8.2). But optimization algorithm will always try to follow path of minimum length while keeping safe clearance from obstacles.

In the given scenario of Figure 8.2, previously developed various versions of HS, DE and their hybrid algorithms have been individually tested as obstacle avoidance strategy in separate runs. Comparison among metaheuristics has been made regarding convergence speed and lowest value of fitness. Control parameters for respective algorithms are specified separately in Table 8.1. Population size, number of decision variables and maximum permissible iterations has been kept fixed for all algorithms mentioned in Table 8.1. Variation in fitness value of respective algorithm has been plotted with respect to iterations in Figure 8.3. Fitness values have chosen as an average of 30 individual runs for each metaheuristics. The variation in fitness value for each algorithm has been recorded separately as an average of their 30 runs.

Proposed Hybrid DE-HS has been compared with significant variants of HS, such as, GHS [131], SGHS [132], ITHS [137] and IGHS [139] during three-dimensional navigation. GHS requires long time iterations to obtain a stable best fitness value comparative to other metaheuristics as shown in Figure 8.3. Underwater robot driven by GHS mechanism may get trapped in local minima situation.

Table 8.1 Details of control parameters for PSO and few variants of HS, DE and their hybrid algorithms applied as path planning algorithm for given situation of Figure 8.2

Algorithms	Details of Control Parameters
GHS [131]:	HMCR=0.9, PAR_{min} = 0.01, PAR_{max}= 0.9.
SGHS [132]:	$HMCR_m$=0.98, PAR_m = 0.9, bw_{min}= 0.0005, $bw_{max} = \frac{UB - LB}{10}$, Learning Period= 100
ITHS [137]:	HMCR=0.99, PAR_{min} = 0, PAR_{max}= 1.
IGHS [139]:	HMCR=0.99, PAR_{min} = 0.01, PAR_{max}= 0.99, bw_{min}= 0.0001, $bw_{max} = \frac{UB - LB}{20}$
PSO [197]:	Inertia constant, ω = 0.729 and c1=c2= 1.494.
JADE [184]:	Mutation strategy: DE/current-to-pbest/1 (with and without archive); $F_i = randc_i(\mu_F, 0.1), \mu_F$ is location parameter of Cauchy distribution and initialized as 0.5; Similarly, $CR_i = randn_i(\mu_{CR}, 0.1), \mu_{CR}$ is mean and initialized as 0.5; c: Constant in Cauchy distribution that controls adaptation rate of F and CR (c=0.1 normally); *p*: Greediness of mutation strategy (*p*=0.05)
SspDE [186]:	A pool of mutation strategies: DE/rand/1/bin, DE/rand to best/2/bin, DE/rand/2/bin, DE/current to rand/1; $F \in (0.1,1)$; $CR \in (0,1)$; Learning Period (LP) = 20 iterations and Reassignment Probability (RP)=0.8

DHS [191]:	HMCR = 0.9, PAR_{min} = 0.01, PAR_{max} = 0.99, Mutation strategy "DE/rand/1/bin" used in pitch adjustment process, CR = 0.9 and F = 0.8.
HSDM[194]:	HMS = 50, HMCR =0.98, PAR is randomly chosen from the set: {0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0}, F= N(0.3,0.5), Mutation strategy contains two pair of differences of randomly chosen vectors from memory.
HIDE [145]:	NFE=220, HMCR= 0.88, PAR_{max} =0.98, and PAR_{min} =0.55, HRT=14, Bw_{min}= 0.2, Bw_{max} = 8, Mutation strategy "DE/rand to best/1", λ = 0.7and F = 0.44.
HSPEADE [198]:	Two mutation strategies: "DE/current-to-pbest/bin" and "DE/current-to-gr best/bin". Both are associated with parameter set: F=0.2 to 1.2, CR=0 to 1, p=0.05 to 2.5, HMCR varies from 0 to 1 and PAR varies from 1 to 0.
Proposed DE-HS:	Mutation strategies and control parameters for pitch adjustment and crossover operations are used as per given details in Section 3.3.

SGHS [132] requires large learning time to adapt control parameters before finding out global best solution. So, average convergence speed of SGHS [132] in the context of present research area has been found to be slow enough in Figure 8.3. ITHS [137] performs a number of stochastic searches within multiple subpopulations simultaneously to enhance the speed of search. Moreover, fitness of best solution derived by ITHS [137] is much lower than other algorithms while finding robot's next best position. In the present scenario, IGHS [139] has shown slow convergence speed initially but increases at final stage of iterations. PSO has also been implemented as three-dimensional navigational strategy using parameter values as specified by Li et al. [197]. But convergence efficiency is not satisfactory compared to other specified metaheuristics.

Among numerous adaptive versions of DE, two important algorithms: JADE [184] and SspDE [186] have been favoured to be implemented in three-dimensional path planning problem. Mechanism of multiple best solutions in JADE facilitates good diversity as well as fast convergence speed as seen in convergence curve of Figure 8.3. SspDE [186] requires a small learning period to assess the list of control parameters and mutation strategies associated with each population members during selection process of DE. Though improvement in fitness of newly generated solution vectors has been observed, still SspDE [186] may face slow convergence speed.

Hybridization of DE and HS as proposed by other researchers has also been implemented as navigational strategies for current scenario for comparison purpose. DHS [191] and HSDM [194] have found to be enough competent to provide lower fitness than other variants of HS like GHS [131], SGHS [132] etc. during navigation. Recently introduced,

HSPEADE [198] algorithm has performed the search for robot's pose more efficiently than DHS [191] and HSDM[194]. Implementation of HIDE [145] in current scenario of three-dimensional navigation has shown effective search quality and good convergence speed. In Figure 8.3, it has been found that IGHS [139], JADE [184], HIDE [145], HSPEADE [198] and proposed DE-HS algorithms have shown better convergence speed along with comparatively lower fitness value than other algorithms for same scenario (Figure 8.2). So, these five algorithms have been selected and implemented as navigational strategies in another simulation scenario (Figure 8.4) with multiple obstacles in between start and goal point. Performances of above mentioned five algorithms have been evaluated regarding deviation in path length from optimum one, total path length to be travelled and also clearance from obstacle (Table 8.2).

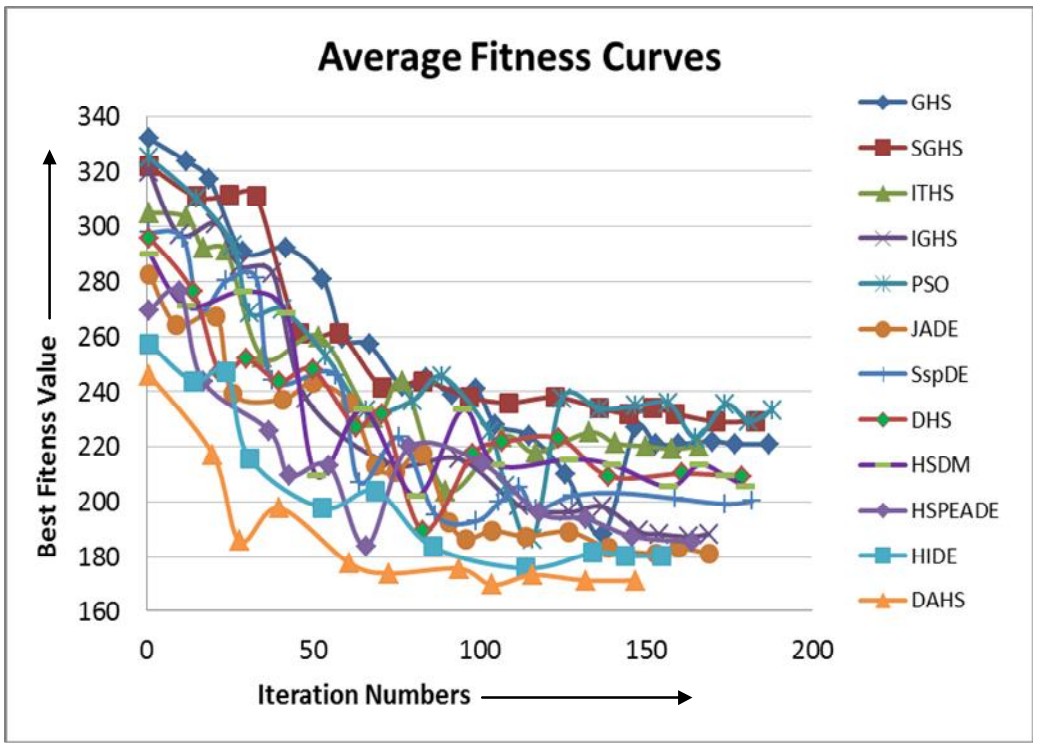


Figure 8.3 Average Fitness Curves for Twelve algorithms while avoiding obstacle in given scenario of 8.2

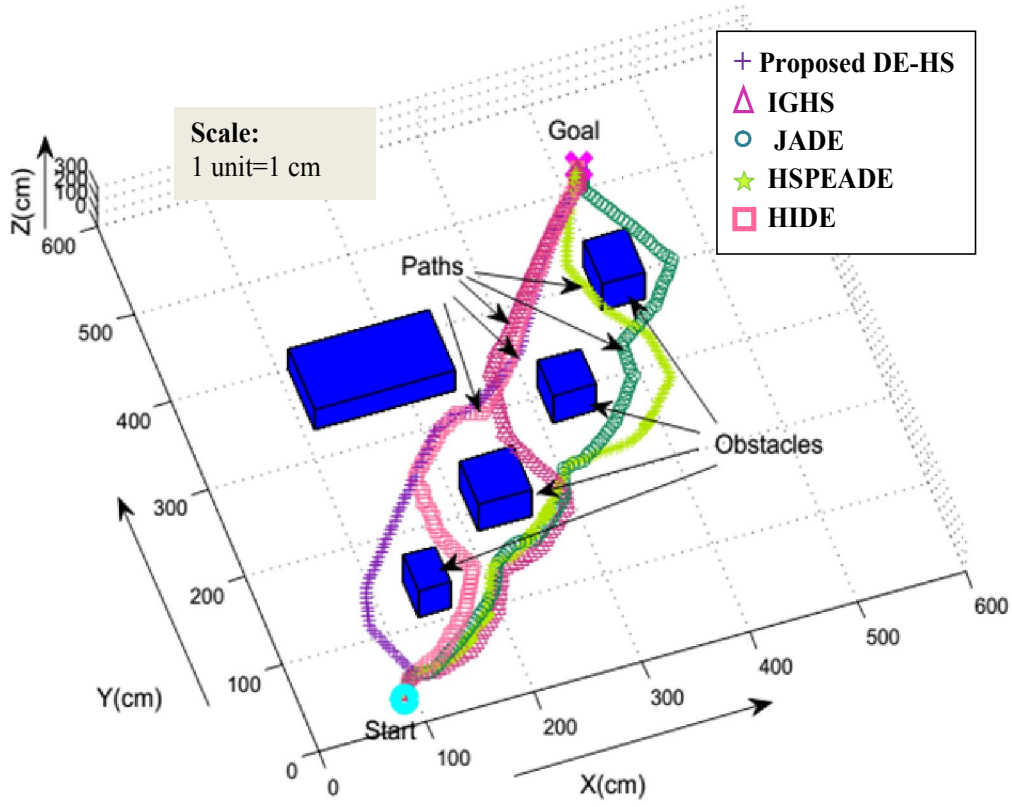


Figure 8.4 Simulated paths traced by five different versions of DE and HS Algorithms

For given scenario, the length of straight line drawn from start to goal has been measured as 432.84 cm. For every algorithm, the same simulation has been performed about 20 times and best path among them has shown in Figure 8.4. Analysing the performances of above mentioned algorithms, proposed Hybrid DE-HS and HIDE [145] algorithms have found to be most effective navigational strategies with reduced path length while avoiding obstacles safely in its way to goal.

Table 8.2 Comparison of proposed hybrid DE-HS with few variants of HS, DE with respect to their performance during navigation

Path Planning Algorithms	Path Length (in cm)	Deviation from Optimum path (in %)	Safe from collision (Yes/No)
IGHS [139]:	491.3	13.52	Yes
JADE[184]:	474.4	9.61	No
HSPEADE[198]:	493.3	13.97	No
HIDE[145]:	466.1	7.68	Yes
Proposed Hybrid DE-HS:	457.5	5.69	Yes

For IGHS [139] and JADE [184], there may be a chance to collide with obstacles as paths traced by them are very close to obstacles as shown in Figure 8.4. HSPEADE has failed to avoid obstacle safely and corresponding path length has also been found to be unsatisfactory in comparison with others. Current comparative study may denote that the proposed hybridization of metaheuristics has better convergence behaviour than other versions of HS, DE and hybrid of DE-HS.

8.5.2 Reactive Behaviors of Underwater Robot for ATHS based navigational strategy

While moving within three-dimensional space, underwater robot has to execute some conventional robotic behaviors such as wall following, obstacle avoidance and target seeking etc. Proposed DE-HS algorithm has been implemented in simulation scenarios of Figure 8.5 and 8.6 with different obstacle arrangements to verify these specified behaviors. As per rules for robot navigation, obstacle avoidance has been prioritized by proposed DE-HS algorithm over target seeking behavior.

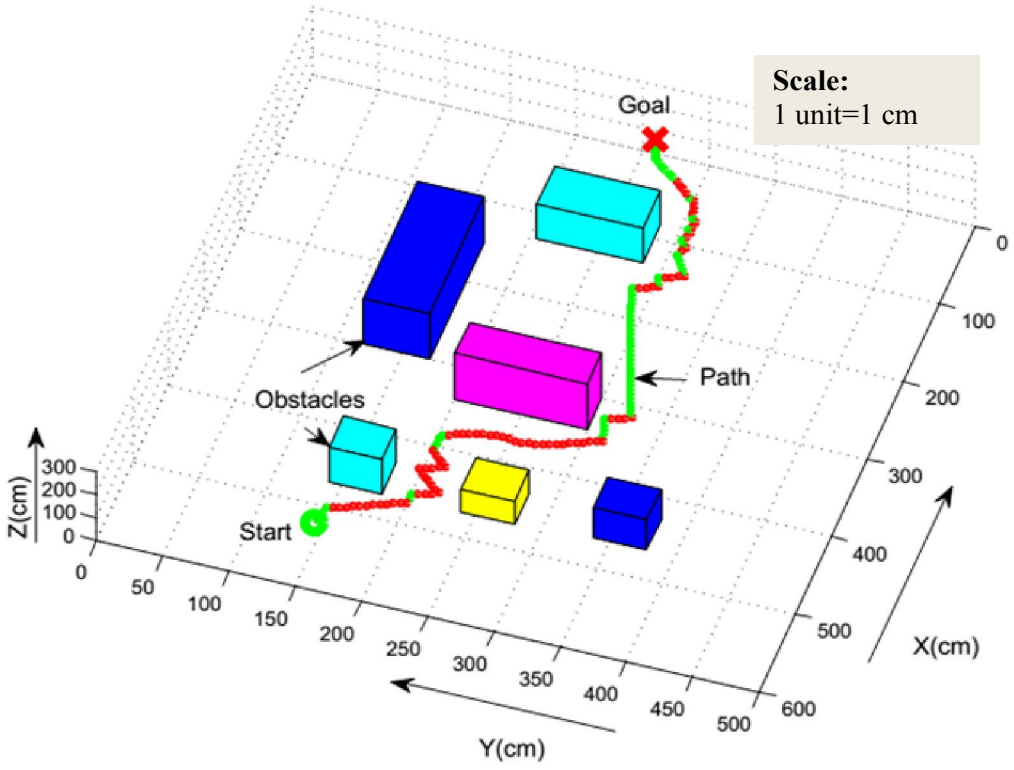


Figure 8.5: Wall following behaviour by proposed DE-HS based navigational strategy

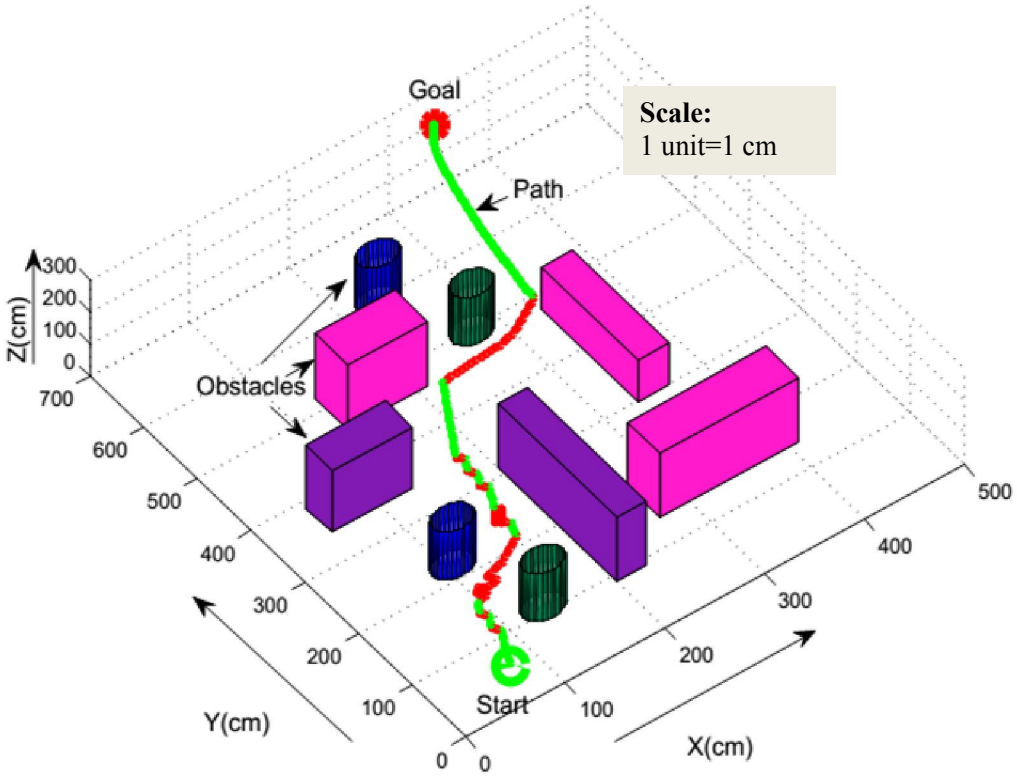


Figure 8.6: Obstacle Avoidance and Target Seeking behaviour by proposed DE-HS

Wall following behavior of proposed navigational strategy has facilitated escaping from any trap like situation or avoiding obstacles of large size during navigation. Both simulation results (Figure 8.5 and 8.6) have shown satisfactory navigational performance of proposed navigational strategy.

8.5.3 Comparison with Other Computational Approaches for Underwater Navigation

In this section other path planning methods for underwater robot have been discussed and compared with the current approach. Proposed navigational strategy has been implemented in the simulated scenarios which contain the same obstacle arrangement along with start and goal positions as used by previous researchers [187,193]. Individual comparisons have been executed here:

(a) In Chapter 6, Navigation of AUV based on Fuzzy logic controller as proposed by Jiang and Zhu [187] has already been considered for executing comparison of navigational performances. The same scenario (Figure 8.7 (a)) has been considered here to implement Proposed DE-HS algorithm as navigational strategy.

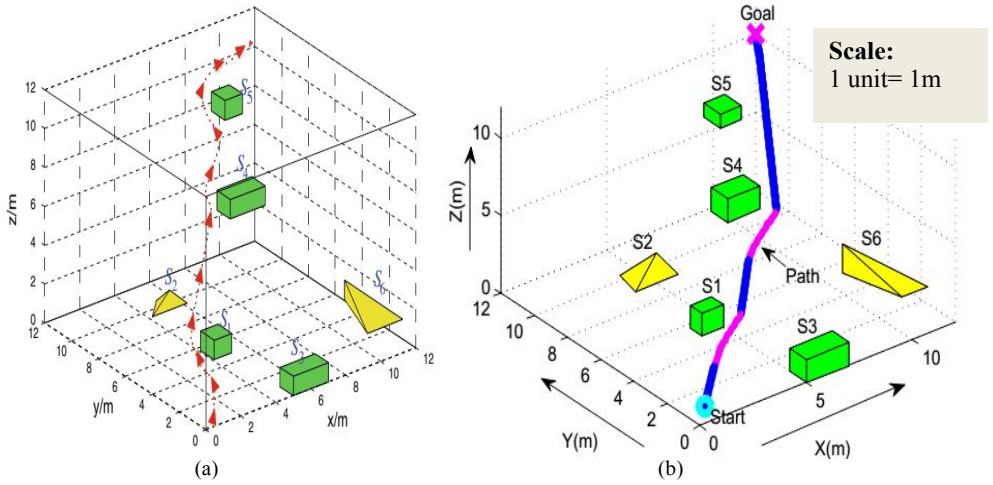


Figure 8.7: (a) Three-dimensional static path planning for AUV using fuzzy logic control by Jiang and Zhu [187]; (b) Simulated path traced by proposed DE-HS approach based navigational strategy

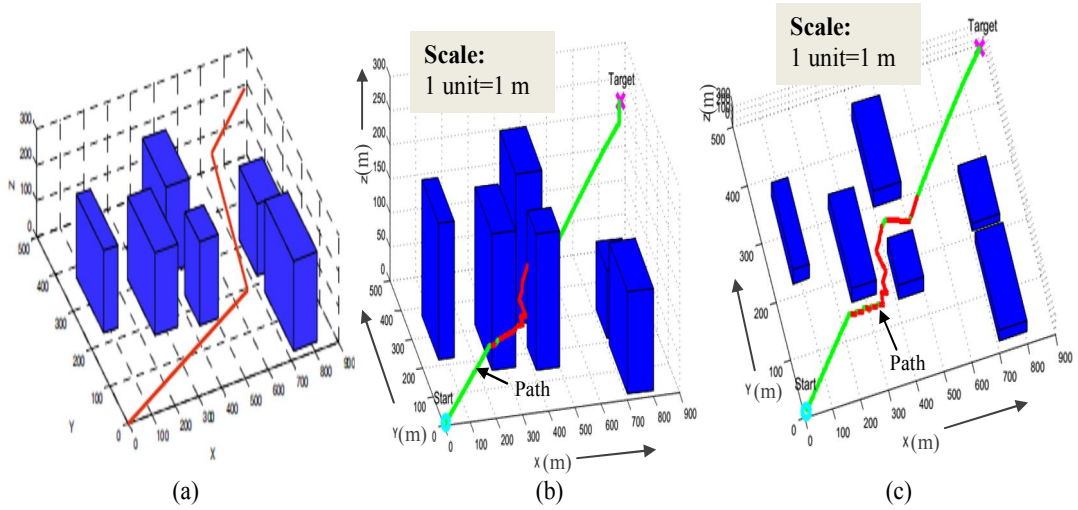


Figure 8.8: (a) 3-D Simulation result using Improved Ant Colony Optimization as shown by Guanglei and Heming [193]; (b) View of Path obtained by implementing proposed DE-HS approach in environment same as Figure 8.8 (a); (c) Another of view of Figure 8.8 (b) at different angle

Figure 8.7(b) has shown the simulated path traced by recent algorithm. Comparison between two approaches regarding path length has been exhibited in Table 8.3. Proposed DE-HS based path planning algorithm has traced shorter path than Fuzzy logic controller

of Jiang and Zhu [187]. Successful avoidance of collision has also been observed during navigation.

(b) Improved Ant colony optimization based navigation of AUV as introduced by Guanglei and Heming [193] has already been discussed in Section 7.5.4 of Chapter 7. Navigational performance of proposed DE-HS algorithm in scenario same as Figure 8.8(a) has been viewed in Figure 8.8 (b). Safe clearance from obstacles has been found in path traced by robot in Figure 8.8 (c) which is another view of Figure 8.8 (b). Comparison between proposed DE-HS and Improved version of ACO algorithm has been depicted in Table 8.3. Recently evolved navigational strategy has presented much better performance than the previous one [193] regarding length and smoothness of path.

Table 8.3 Comparison between Proposed DE-HS algorithm and other navigational strategies for navigational performance within simulated scenarios

Figure No.	Navigational Strategy	Length of 3D path traced by underwater robot (in cm)	Deviation (in %)
8.7(a)	Fuzzy logic based approach [187]	223.9	4.85
8.7(b)	Proposed DE-HS approach	213.5	
8.8(a)	Improved ACO [193]	1087.8	2.08
8.8(b)	Proposed DE-HS approach	1076.7	

8.6 Experimental Results

In this section, simulated performance of proposed DE-HS based navigational strategy has been verified in real world environment along with utmost accuracy. Figure 8.9 has shown experimental views of underwater robot "GNOM-Baby" at different stages of navigation. Obstacle positions in experimental scenario of Figure 8.9 have been located by following the simulation scenario of Figure 8.6. In experimental mode, proposed DE-HS algorithm based navigational strategy has successfully avoided obstacles while moving from start to goal position as viewed in Figure 8.9.

8. 6.1 Comparison between simulation and experimental results

Both simulation and experimental results have individually executed for 20 times. In each run, path length and time taken for both simulation and experimental results have been recorded in Table 8.4 and 8.5 respectively. The difference between actual and optimum path length has been derived for both simulation and experimental results. The optimum

distance between start and goal positions has been recorded as 641.6 cm and 678.7 cm for simulation (Figure 8.6) and experimental (Figure 8.9) scenarios respectively.

Comparison between simulation and experimental results has been carried out in terms of path length and time taken in Table 8.4 and 8.5 respectively. Average of differences between simulated and experimental path length has been found to be significantly low in Table 8.4. Comparison regarding travel time has also shown good agreement in Table 8.5. So, performance of DE-HS based navigational strategy can be considered as robust one during underwater navigation.

Table 8.4 Comparison between experimental and simulated results for path length in Scenario1 (Figure 8.6 and 8.9)

1	2	3	4	5	6
No. of Runs	**Path length in simulated mode (cm)**	**Deviation from Optimum path (%)**	**Path length in experimental mode (cm)**	**Deviation from Optimum path (%)**	**Difference between column 2 and 4 (%)**
1st	673.2	4.93	706.4	5.15	4.58
2nd	678.8	5.80	713.2	6.16	5.99
3rd	676.3	5.42	716.3	6.63	5.78
4th	671.7	4.69	703.7	4.76	5.83
5th	667.2	3.99	701.9	4.49	5.51
6th	679.1	5.86	718.5	6.96	6.53
7th	657.8	2.53	696.5	3.68	6.49
8th	668.5	4.20	702.9	4.63	5.88
9th	675.3	5.26	709.7	5.64	5.82
10th	669.4	4.34	701.4	4.41	5.45
11th	671.9	4.73	704.7	4.90	5.16
12th	667.9	4.10	695.9	3.59	6.83
13th	673.3	4.94	709.3	5.59	5.25
14th	670.2	4.46	707.2	5.27	4.71
15th	675.5	5.29	711.5	5.91	4.38
16th	678	5.68	713	6.14	5.03
17th	673.7	5.01	710.9	5.83	4.78
18th	677.3	5.57	702.3	4.54	5.22
19th	674.8	5.18	704.8	4.92	5.50
20th	672.3	4.79	712.2	6.03	6.70
Average difference in path length between simulated and experimental results					5.13

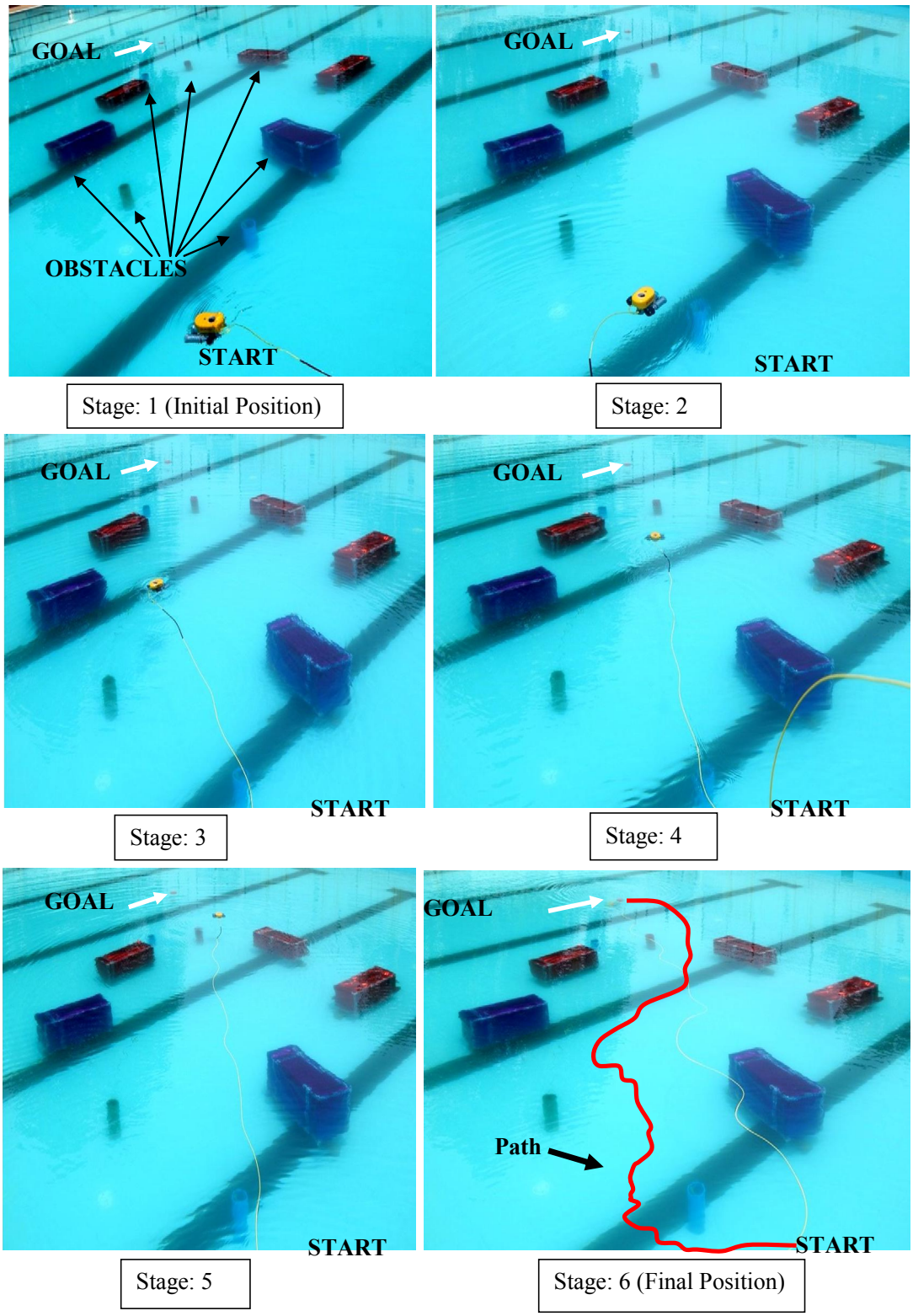


Figure 8.9: Experimental view for proposed DE-HS based underwater navigation of GNOM Baby for obstacle arrangement near about same as in Figure 8.6

Comparative study has been extended for a number of navigational scenarios which have been created in both simulation and experimental modes. Navigational behavior of proposed strategy has been observed for all scenarios. Among numerous testing of each scenario, best path length and minimum travel time has been recorded in Table 8.6 and 8.7 respectively. Average deviation between simulated and experimental performances of proposed navigational strategy has not been much varied for the change in scenario. Therefore, proposed hybrid DE-HS based path planning algorithm can trace collision free path within a chaotic underwater environment.

Table 8.5 Comparison between experimental and simulated results for travel time in Scenario1 (Figure 8.6 and 8.9)

1	**2**	**3**	**4**
No. of Runs	**Travel time in simulated mode (in sec)**	**Travel time in experimental mode (in sec)**	**Difference between column 2 and 4 (in %)**
1st	24.04	25.50	6.07
2nd	16.16	16.86	4.32
3rd	20.50	21.64	5.59
4th	18.15	18.97	4.4%
5th	23.01	24.35	5.83
6th	17.41	18.22	4.65
7th	14.62	15.31	4.72
8th	18.57	19.47	4.86
9th	17.98	18.82	4.70
10th	20.92	21.98	5.08
11th	17.27	18.13	4.99
12th	15.61	16.13	3.32
13th	19.80	20.84	5.25
14th	20.94	22.08	5.42
15th	25.02	26.48	5.84
16th	19.83	20.83	5.04%
17th	21.73	22.93	5.49
18th	23.53	24.62	4.60
19th	21.20	22.07	4.12
20th	14.94	15.78	5.64
Average difference in travel time between simulated and experimental results			5.02

Table 8.6 Comparisons between experimental and simulated results for five more scenarios regarding path length

Different Scenarios	Best path length in Simulation (in cm)	Best path length during Experiment (in cm)	Difference (in %)
Scenario-2	1045.3	1103.2	5.53
Scenario-3	732.5	771.4	5.31
Scenario-4	861.9	907.5	5.28
Scenario-5	927.4	974.6	5.09
Scenario-6	758.2	797.4	5.16

Table 8.7 Comparisons between experimental and simulated results for five more scenarios regarding travel time

Different Scenarios	Minimum travel time during simulation (in sec)	Minimum travel time during experiment (in sec)	Difference (in %)
Scenario-2	28.25	29.68	5.07
Scenario-3	21.55	22.68	5.28
Scenario-4	23.30	24.46	5
Scenario-5	26.50	27.89	5.24
Scenario-6	21.66	22.72	4.86

8.7 Summary

Two simple metaheuristics, DE and HS, which are complementary for each other, have been integrated here in a cooperative manner to enhance the convergence speed and search quality of three-dimensional path optimization process. Simulation and experimental investigations on proposed hybrid metaheuristics based navigational methodology have been executed in a realistic manner and can be analysed as follows:

- Initially, proposed hybrid algorithm has followed the optimization process of stochastic nature to explore the search space in a broad manner. Global optimization process has successfully identified the region of search space which may contain the global optimal solution. With the advance of iteration numbers, deterministic nature has been incorporated in search process to finely tune the vectors which are near about the global optimal solution.

- Automatic selection of mutation strategy during pitch adjustment process, DE's crossover operation and online updating mechanism for control parameters are the significant features of proposed DE-HS algorithm.

- Hybridization of metaheuristics (DE-HS) has improved the convergence behaviour by reducing computational complexity in comparison with standard algorithms (HS or DE). During optimization cycle, fastest convergence speed and lowest fitness value has been recorded for proposed DE-HS algorithm.

- The feasibility of proposed DE-HS algorithm has been verified to trace in collision free near optimal path for underwater robot numerous simulated scenarios with different obstacle arrangements.

- Comparison with other path planning algorithms has evaluated efficacy of proposed DE-HS algorithm during underwater navigation.

- Experimental verifications of simulated performances have validated the proposed DE-HS algorithm as a three-dimensional navigational strategy. Average difference has been computed between simulated and experimental results regarding path length (5.25%) and travel time (5.08%) which can prove the robustness of proposed DE-HS based navigational strategy.

Present chapter has effectively blended two different heuristic search methods and also implemented the proposed hybridization as navigational strategy of underwater robot. Less computational time, higher degree of optimality in path selection and local minima avoidance etc. have been considered as objectives of performed hybridization process. Analyzing simulated and experimental performances of proposed DE-HS based navigational method, it can be stated that all the objectives has been achieved in a satisfactory manner. In next chapter, one more trial on hybrid approach will be performed for achieving desirable navigational performance of underwater robot.

9. Hybrid Learning Approach for ANFIS based Navigational Strategy of Underwater Robot

The computational complexity and run time of navigational algorithm has been tried to reduce by using only two ANFIS models. Learning algorithm of ANFIS has been modified by employing metaheuristic approach like adaptive SFLA. The objective is to reduce the prediction error.

9.1 Introduction

Performance of ANFIS system depends on several internal parameters of the system, such as number of input variables and its membership functions, parameters for defining shape of membership function, parameters in consequent part to accurately scale the input variables, number of training patterns, the iterations or epochs required for training and most importantly the permissible error between actual predicted outputs [199]. Selection of input variables and its membership functions can be done based on heuristic knowledge acquired from experts or using system model developed by numerous input-output training data pairs. Mostly, both heuristic and model based approaches are blended in ANFIS model design to obtain actual output which is near about its predicted value [200]. The adaptive capability of ANFIS makes it possible for immediate adaptive and learning control. This adaptive network has good ability and performance in system identification, prediction and control and has been applied in many different systems. ANFIS has the advantage of good applicability as it can be interpreted as local linearization modelling and conventional linear techniques for state estimation and control are directly applicable [201]. First, it uses the training data set to build the fuzzy system in which, membership functions are adjusted using the back propagation algorithm, allowing that the system learns with the data that it is modelling. Two ANFIS models have been coupled together to estimate required changes in heading angles of underwater robot {($\Delta\Psi$ and $\Delta\theta$} in horizontal and vertical planes respectively.

9.2 Simplified ANFIS Architecture

During navigation, online information from on board sensors of underwater robot such as distances of nearest obstacle (NOD) and target (TD) from robot, heading angles between robot and its nearest obstacles (angobs(h) and angobs(v)) in the horizontal and vertical

planes respectively and target angles both in horizontal (angtar(h)) and vertical (angtar(v)) planes have been fed to the proposed ANFIS structure (as developed in Figure 9.1). ANFIS model1 has considered four inputs only (NOD, TD, angobs(h) and angtar(h)} to provide the required change in heading angle of underwater robot (ΔΨ) in horizontal plane.

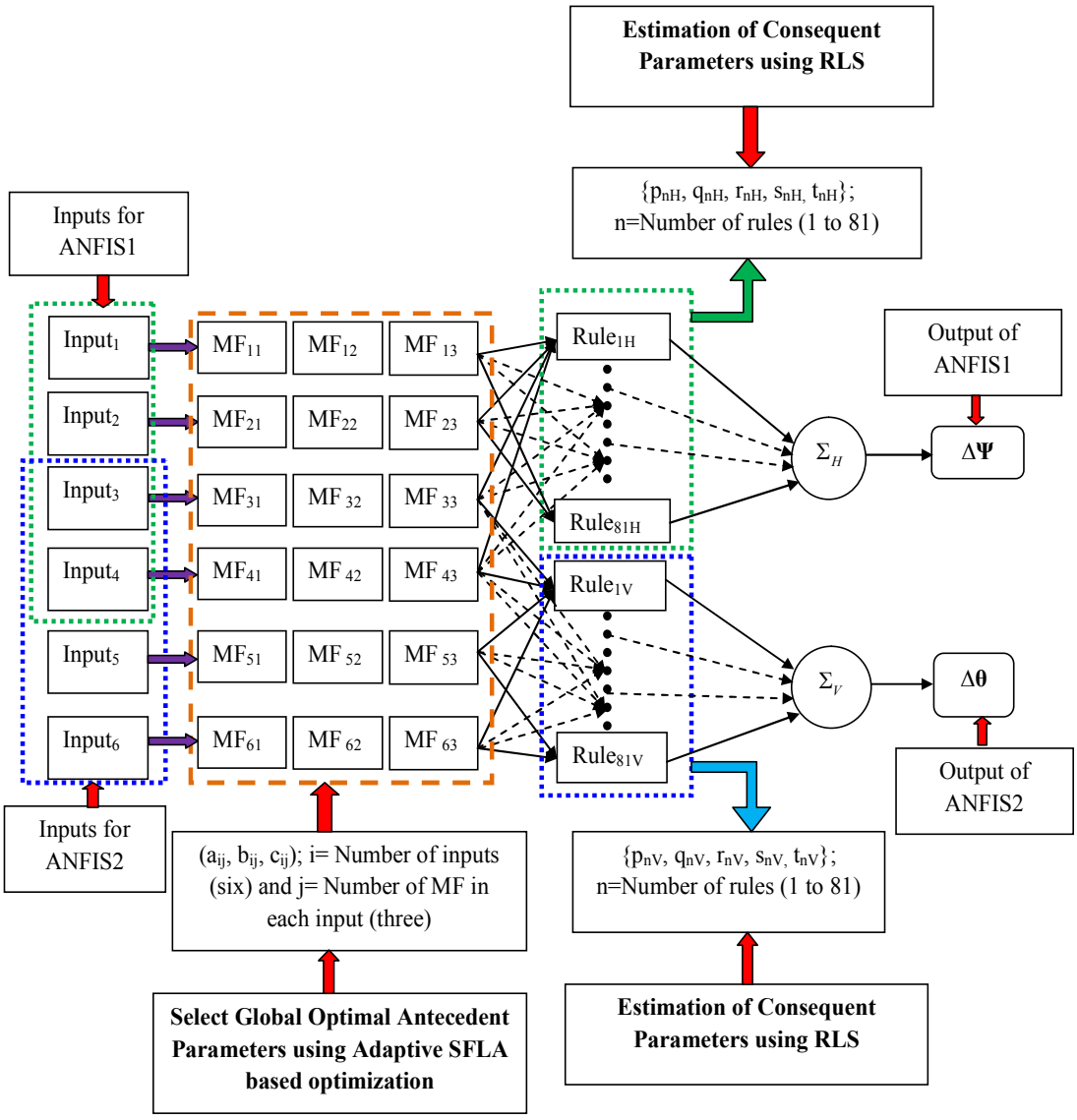


Figure 9.1: Hierarchical Structure of Two ANFIS models with Parameters to be optimized

Similarly, four inputs (NOD, TD, angobs (v) and angtar (v)} have been assigned for ANFIS model2 which estimates the required change in heading angle of underwater robot (Δθ) in vertical plane. Coupling between two ANFIS models (Figure 9.1) has found to be

beneficial for regulating the motion direction of underwater robot. Same as Chapter 4, three bell-shape membership functions have been used for each input variables. Linguistic terms and distribution for membership functions have already been described in Chapter 4.

Table 9.1:

(a) Rules for ANFIS model1 to estimate change in heading angle within horizontal plane

No. of Rules	Robotic Behaviour	Input Variables				$\Delta\Psi$
		NOD	TD	angobs(h)	angtar(h)	
(i)	OA	Near	Far	RCW	Not considered	RCCW
(ii)	TS	Far	Medium	RCCW	NR	NR
(iii)	OA	Medium	Far	RCCW	Not considered	RCW
(iv)	OA&TS	Near	Medium	RCW	RCW	RCW
(v)	TS	Medium	Medium	RCCW	RCW	RCW
(vi)	TS	Far	Near	RCW	NR	NR
(vii)	OA	Medium	Far	NR	Not considered	RCW
(viii)	OA&TS	Near	Medium	NR	RCCW	RCCW
(ix)	OA&TS	Medium	Medium	RCW	RCW	RCW

(b) Rules for ANFIS model2 to estimate change in heading angle within vertical plane

No. of Rules	Robotic Behaviour	Input Variables				$\Delta\theta$
		NOD	TD	angobs(h)	angtar(h)	
(i)	OA&TS	Near	Medium	NR	RCCW	RCCW
(ii)	OA	Near	Far	RCW	Not considered	NR
(iii)	TS	Far	Medium	RCCW	RCW	RCW
(iv)	OA&TS	Near	Near	RCW	NR	NR
(v)	TS	Medium	Medium	NR	RCCW	RCCW
(vi)	OA	Medium	Far	NR	Not considered	RCW
(vii)	OA	Near	Far	RCCW	Not considered	RCW
(viii)	OA&TS	Near	Medium	RCCW	RCCW	NR
(ix)	OA&TS	Medium	Medium	RRCW	RCW	RCW

Note: OA: Obstacle Avoidance, TS: Target Seeking

Rule-base of each ANFIS model which has been designed for four inputs and one output contains 81 rules to represent all possible combination of the membership functions in input variables. Therefore, rule base of proposed ANFIS architecture have composed of (81x2) 162 rules. Few rules of both ANFIS models can be viewed in Table 9.1. For example, rule 8 has been proposed for a certain position of underwater robot where, a static obstacle has been which is near to robot and located at left side of robot orientation of Positive and target is found at Far with angle denoted as negative. Output of the rule in T-S model based FIS will provide crisp value of required change in heading angle of underwater robot to avoid the detected obstacle successfully in its way to goal.

9.3 Learning algorithms for ANFIS

Performance of ANFIS depends on several internal parameters of the system, such as number of input variables and its membership functions, parameters for defining shape of membership function, parameters in consequent part to scale the crisp values of input variables, number of training patterns, the iterations or epochs required for training and most importantly the permissible error between actual predicted outputs. As rules of FIS are a function of input variables and their respective membership functions, the size of rule-base will be effectively influenced by any change in premise parameters. Moreover, if the crisp value of input variables is not properly scaled using consequent parameters, then erroneous output may be produced. Therefore, effective learning algorithm is required to tune the parameters of both antecedent and consequent parts of rules in developed ANFIS architecture [199].

Selection of parameters of fuzzy membership functions based on trial-and-error approach may provide a suboptimal parameter set. The most conventional learning scheme for ANFIS is back propagation algorithm. Such derivative based approach executes local search which solely depends on initial value of learning rate and momentum parameters of the algorithm. But, computation of derivatives by following chain rule may lead towards local minima. Therefore, BP algorithm cannot be treated as global optimization and its convergence speed has also found to be slow enough. Jang [77] has combined the gradient descent method and the least squares estimate (LSE) to improve the learning speed of ANFIS model. Antecedent parameters of each rule have been optimized through GD method and corresponding consequent parameters are estimated using LSE method.

Though GD+LSE based training algorithm has been widely applied, still its convergence efficiency has remained at below average.

In past few years, several evolutionary based global search approaches artificial bee colony (ABC) [203], particle swarm optimization (PSO) [199], Differential Evolution (DE) [203], and ant colony optimization (ACO) [204] etc. have been employed as training algorithm for ANFIS network for eliminating drawbacks of BP algorithm. Population based stochastic approaches can avoid local minima to some extent while minimizing computational time. Such derivative-free global optimization can enhance the accuracy in parameter estimation without affecting the ANFIS's human like ability to deal with ambiguities of the real world problems. However, with the increase of complexity in ANFIS structure, stochastic optimization methods may converge slowly as it requires large memory and long processing time to train a large number of antecedent and consequent parameters. Moreover, local minima situation may also occur during global search for best parameter set for developed ANFIS Model. To avoid such drawbacks, the concept of hybrid learning approach has already been introduced to train the ANFIS parameters with high degree of accuracy. In hybrid training mechanism, usually, heuristic approach based optimization has been employed to train premise parameters and consequent parameters have simultaneously updated through derivative based computation. Apart from gradient descent approach, Least square estimation (LSE) [77], recursive LSE (RLS) [205] or Kalman filter based estimation [199] have also been utilized by many researchers to accurately determine the consequent parameters. But their performances are specific to the number of input variables and parameters of membership functions. However, available learning algorithms for ANFIS may not ensure the optimal choice of parameters irrespective of any constraints.

To get rid of such complexities in training process for ANFIS, proposed adaptive version of shuffled frog leaping algorithm (in Chapter 5) has been introduced here to tune parameters of membership functions more accurately than other approaches and RLSE has been employed to estimate the consequent parameters. Being independent of learning rate, the SFLA can enhance convergence speed and search quality while searching best parameter set for ANFIS model.

9.4 Proposed hybrid learning scheme based on SFLA and RLS method

The proposed combinatorial learning algorithm for ANFIS model has followed the structure of the hybrid training approach as introduced by Jang [77]. Therefore, it can be represented as a combination of two steps: forward pass and backward pass. In forward pass, consequent parameters of rules have been identified through recursive least square estimation (RLS) scheme for constant value of premise parameters. In backward pass, proposed adaptive SFLA based optimization has been used to adjust the premise parameters of ANFIS model while the consequent parameters will be fixed. All premise and consequent parameters of the proposed ANFIS architecture which are required to be trained have been depicted in Figure 9.1. The error between actual and predicted outputs, which will be propagated in backward direction of ANFIS layers, will be treated as fitness value in SFLA based optimization. Proposed training process has been described in next subsections:

9. 4.1 RLS based parameter estimation in forward pass:

In the forward pass, inputs of training patterns are fed into the Fuzzification layer of ANFIS architecture associated with the fixed premise parameters. Here, the objective is to identify the consequent parameters of rules for minimizing the error between estimated and predicted output of ANFIS. Each rule contains five consequent parameters $\{p_0, p_1, p_2, p_3, p_4\}$ which are linearly combined with crisp value of four input variables as shown in Figure 9.1. Number of rules in one ANFIS system is $3^4 = 81$. So, (81x5) 405 numbers of consequent parameters are required to be estimated in each ANFIS model. As proposed architecture has two ANFIS models, the total (405x2) 810 consequent parameters will be involved in training process. Least square estimation may require huge computational power and long processing time to deal with such large data. But recursive least square estimation (RLS) can calculate numerous complex parameters with much more ease and within lesser run time than LSE. In the literature, solution estimated by RLS method has also been found to be more reliable than least square estimation [205]. The RLS based estimation for parameters has been illustrated as follows:

If the set of input variables is $Q = (x_1, x_2, ..., x_i)^T$ and the predicted outputs within available training data set $T_D = (T_1, T_2, ..., T_N)$ has been represented as $Y_D = (y_{D1}, y_{D2}, ..., y_{DN})$, then the estimated output of ANFIS model can be formulised for t^{th} training pattern in a generalized manner:

$$\hat{y}_{D,t} = \sum_{h=1}^{n} \bar{w}_{D,t}^{h} f_t^h = \sum_{h=1}^{n} \bar{w}_{D,t}^{h} \left(p_0^h + p_1^h x_{1t} + ,\ldots\ldots\ldots, + p_i^h x_{it} \right)$$
$$= \left[\bar{w}_{D,t}^{1}, \bar{w}_{D,t}^{2}, \ldots\ldots, \bar{w}_{D,t}^{n}, \bar{w}_{D,t}^{1} x_{1t}, \bar{w}_{D,t}^{2} x_{1t}, \ldots\ldots\ldots, \bar{w}_{D,t}^{n} x_{1t}, \ldots\ldots, \bar{w}_{D,t}^{1} x_{qt}, \bar{w}_{D,t}^{2} x_{it}, \ldots\ldots\ldots\ldots, \bar{w}_{D,t}^{n} x_{it} \right]$$
$$\times \left[p_0^1, p_0^2, \ldots\ldots\ldots p_0^n, p_1^1, p_1^2, \ldots\ldots\ldots\ldots, p_1^n, \ldots\ldots\ldots, p_i^1, p_i^2, \ldots\ldots\ldots, p_i^n \right]$$

(9.1)

Where, h: no. of rules; t: no. of training patterns; i: no. of input variables; D: stands for H (Horizontal) or V(Vertical) plane; $\bar{w}^h$: Normalized firing strength for h^{th} rule; f^h : Output for h^{th} rule of T-S based fuzzy model which is linear combination of consequent parameters $\{p_0, p_{1, \ldots\ldots\ldots}, p_i\}$ and set of input variables $Q = (x_1, x_2, \ldots, x_i)^T$.

eq. 9.2 can be represented in matrix form of estimated output with respect to input parameters of N training data set, as follows:

$$\hat{Y}_D = S_D^T X_D \tag{9.2}$$

Where, $\hat{Y}_D = [\hat{y}_{1,D}, \hat{y}_{2,D}, \ldots\ldots, \hat{y}_{N,D}]$: Actual estimated output of ANFIS model for horizontal or vertical plane; $S_D = \left[p_0^1, p_0^2, \ldots\ldots\ldots p_0^n, p_1^1, p_1^2, \ldots\ldots\ldots\ldots, p_1^n, \ldots\ldots\ldots, p_i^1, p_i^2, \ldots\ldots\ldots, p_i^n \right]$: Consequent Coefficients in vector form and

$$X_D = \begin{bmatrix} \bar{w}_{D,1}^{1}, .., \bar{w}_{D,1}^{n} & \bar{w}_{D,1}^{1} x_{11}, .., \bar{w}_{D,1}^{n} x_{11} & ..... & \bar{w}_{D,1}^{1} x_{q1}, .., \bar{w}_{D,1}^{n} x_{q1} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ \bar{w}_{D,N}^{1}, .., \bar{w}_{D,N}^{n} & \bar{w}_{D,N}^{1} x_{1N}, .., \bar{w}_{D,N}^{n} x_{1N} & ..... & \bar{w}_{D,N}^{1} x_{qN}, .., \bar{w}_{D,N}^{n} x_{qN} \end{bmatrix}$$: Combination of input variables and firing strength for 'n' rules in a vector form.

In current iteration, the error between actual and predicted output for given training patterns can be defined as, $er(i) = Y_D(i) - \hat{Y}_D(i) = Y_D(i) - S_D^T(i) X_D(i)$ (9.3)

To determine the consequent coefficients, one cost function has been chosen as square of prediction error defined in eq. 9.3. If error is in direct proportion with consequent parameters, then, the optimized value of coefficients will enhance the speed of error minimization. The matrix of estimated consequent parameters at i^{th} iteration can be defined in a recursive form,

$$S_D(i) = S_D(i-1) + P_D(i)X_D(i)(Y_D(i) - S_D^T(i)X_D(i)) \quad (9.4)$$

Where, Covariance matrix P(i) can also be found out in a recursive manner by following an Algebraic Riccati equation which is parallel to Kalman filter,

$$P_D(i) = P_D(i-1) - P_D(i-1)X_D(i)(I + X_D^T(i)P_D(i-1)X_D(i))^{-1}X_D^T(i)P_D(i-1) \quad (9.5)$$

For each training pattern, the recursive form of least square method is able to update the matrix of the consequent coefficients, $S_D(i)$ irrespective of previous values of parameters. Here, the computational liabilities of the inverse matrixes have also been reduced than LSE. Hence, estimated solution using RLS can be considered as reasonable, consistent and cost-effective.

9. 4.2 Tuning of premise parameters using SFLA in backward pass:

In backward pass, premise parameters have been tuned through adaptive SFLA algorithm for fixed value of consequent parameters as identified in forward pass. By following the proposed architecture (Figure 9.1), (6 x 3) 18 membership functions are required to be defined for representing input variables. Each bell-shape membership function contains three parameters {a_{ij}, b_{ij}, c_{ij}; where, i: number of inputs and j: number of parameters in membership function (1 to 3)}. So, population of Adaptive SFLA has been specified as a set of solution vectors which are representation of all possible sets (18 x 3 = 54) of real-valued antecedent parameters as shown in Figure 9.1. Formulation of Adaptive SFLA based metaheuristic approach has been illustrated in Section 5.3 of Chapter 5. Population size and distribution of solution vectors within memplexes have been decided based on trial and error analysis as shown in Section 9.5 of current chapter. Population of Adaptive SFLA for six inputs can be represented as a matrix of 150x54 dimensions which has been declared and randomly initialized in a Microsoft excel sheet. MATLAB code can easily access that excel file to initiate optimization process of SFLA. Accuracy of estimation through ANFIS model highly depends on the control parameters of training algorithm. So, other parameters mentioned in SFLA's Pseudo code (as given in Section 5.3) have been chosen through extensive trial and error analysis such as: No. of maximum iterations or no. of shuffling iterations for SFLA, is_{max} =30, No. of decision variable for each solution vectors of population: 3 as only three parameters are required to define bell-shape membership function, No. of maximum evolution within each submemplex, im^{max}

=10. The fitness of each antecedent parameter set has been determined here in terms of RMSE value of ANFIS model with respect to training data set. RMSE can be estimated as follows:

$$RMSE = \sqrt{\frac{1}{t}\sum_{t=1}^{T}(y_{t,D} - \hat{y}_{t,D})} \tag{9.6}$$

Where, T is the total number of available training patterns; D: (H or V) denotes horizontal or vertical plane; $y_{t,D}$ is the heading angle in horizontal or vertical plane as predicted in training data set, $\hat{y}_{t,D}$ is the value of heading angle in horizontal or vertical plane as estimated by proposed SFLA-RLS based hybrid learning of ANFIS model.

SFLA has been employed in an iterative manner to minimize RMSE of ANFIS model by optimizing combinations of antecedent parameters by following steps as given in next sub-section. With the progress of iterations in SFLA, sets of antecedent parameters will be evolved. If the fitness or RMSE value is less than its specified threshold then SFLA iterations will be stopped and solution vector associated with lowest RMSE will be considered as best fitted set of premise parameters.

9. 4.3 Steps of hybrid learning method for ANFIS architecture

Flow chart for proposed hybrid learning scheme for ANFIS model has been drawn in Figure 9.2 and each step of algorithm has been briefly discussed here in a sequential manner:

Step I: *Initialization*

Before initiating the training process for ANFIS models, control parameters for SFLA must be initialized to certain values which have been found to be suitable for present application in trial and error analysis. Each solution vector in population of SFLA must represent a set of antecedent parameters. Dimension of population has already been mentioned in previous subsection. SFLA population has been randomly initialized within the specified ranges of membership parameters. Desirable lowest value of fitness function or RMSE has to be declared in this step.

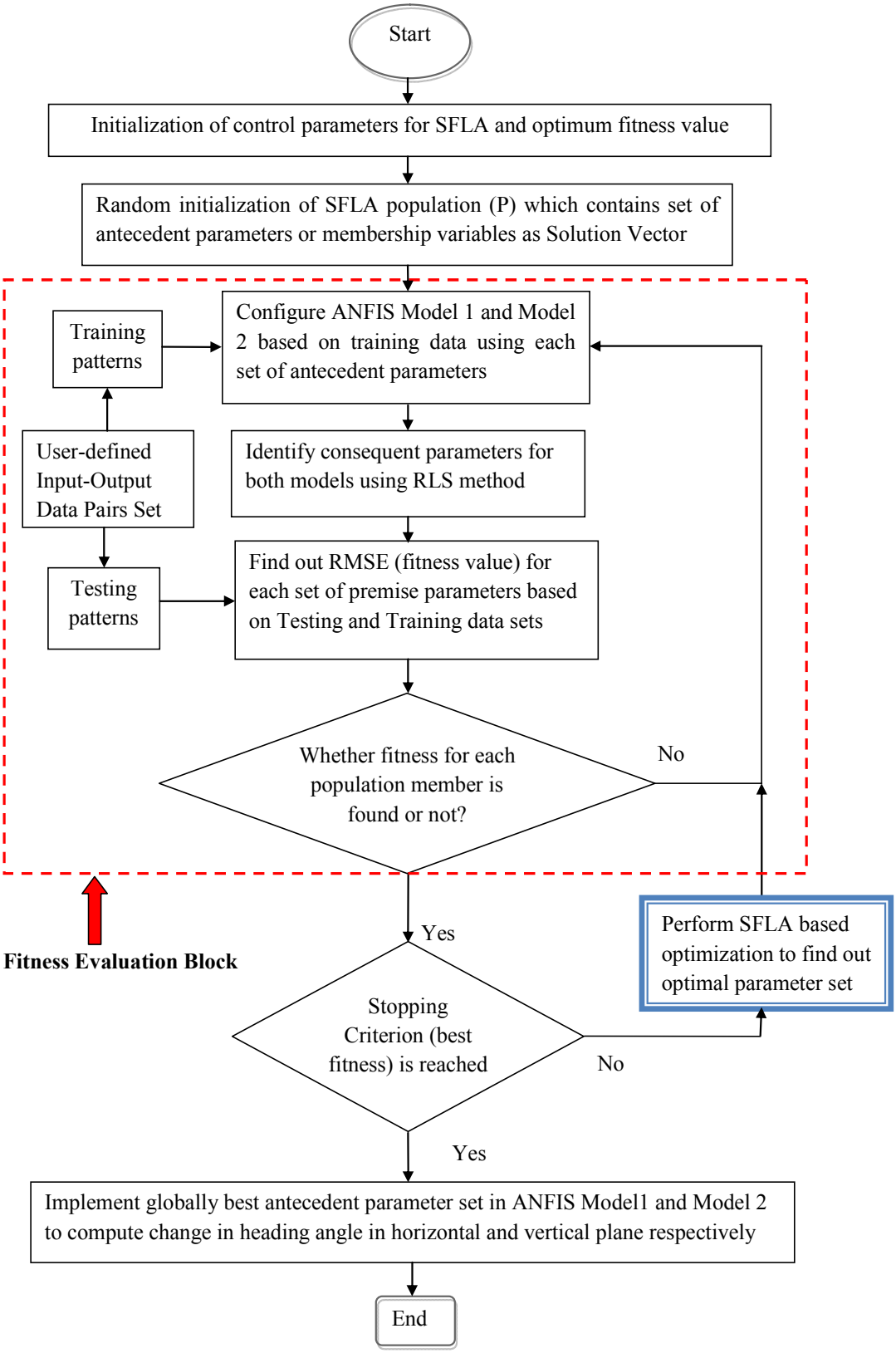


Figure 9.2: Flowchart for training of ANFIS architecture based on proposed SFLA –RLS based hybrid learning approach

Step II: *Fitness Function Evaluation*

For each set of antecedent parameters, six layered architecture of ANFIS model 1 & 2 (as shown in Figure 9.1) will be configured based on training data set of 350 training patterns. RLS method has been used to identify consequent parameters of corresponding antecedent parameter set by following the derivation given in sub-section 9.4.1. The performance of configured architecture will also be verified based on testing data set of 150 testing patterns. RMSE value for each solution vector has been computed by utilising eq. 9.6 and stored in an array.

Step III: *Stopping Criterion*

Fitness array will be sorted to find out lowest fitness value. If the lowest fitness of array is almost same as specified threshold value for fitness, then iterations will be stopped and set of antecedent parameters associated with best fitness will be considered as optimal setting for proposed ANFIS architecture. In other way, if lowest fitness or RMSE in array has remained constant for consecutive iterations, then iterations will come to an end. Corresponding set of antecedent parameters will be treated as best combination of antecedent parameters for two ANFIS models in current application to compute near optimal change in steering angle. If stopping criterion is not satisfied then go to step IV to optimize the current population.

Step IV: *Training using SFLA based optimization*

Population of sets of antecedent parameters will be evolved through modified version of SFLA based optimization as described in Section 5.3 which perfectly combines subpopulation concept for local search and shuffling strategy for global search. Within local search process of SFLA, new evolution strategy has been added in eq. 5.6 of Chapter 5 to perfectly tune the solution vector for avoiding the premature convergences. Fitness evaluation of each population member of SFLA has been performed by following step II for all given training patterns. As entire population evaluated in one iteration of training algorithm, therefore, the total number of required evaluations in SFLA can be determined as (population size x specified maximum iteration of SFLA). After completion of final iteration of SFLA, stopping criterion must be checked whether satisfied or not? If not satisfied, then SFLA optimization will start again, otherwise step III will be followed.

Steps II to IV will be repeated in a cycle until the optimal parameters for fuzzy membership functions has not been found by SFLA. Final values of consequent parameters with respect to recently optimized value of antecedent parameters will be determined by using RLS method.

9.5 Analysis on Performance of SFLA-RLS based hybrid learning approach

Performance of the proposed training algorithm based on hybridization of SFLA and RLS can be evaluated in terms of differences between actual outputs of developed ANFIS structure and predicted outputs from training or testing data set, number of training patterns required for accurate parameter estimation, convergence speed, number of epochs required to converge, lowest value of RMSE during training and testing periods and CPU usage time. Moreover, optimization behaviour of metaheuristic like SFLA has been highly influenced by its population size. A quantitative analysis on SFLA based optimization of premise parameters has been here performed by plotting the RMSEs or finesses of optimal solutions with respect to variation in population size. A subsequent decrease in RMSE value has been recorded with the increase in population size (Figure 9.3). But computation time required for optimization will be significantly enhanced for rise in population size. Therefore, a proper trade-off between population size and run time for algorithm must be found out. It has also been observed that after a certain limit, no change has been occurred in RMSE value for any increase in population size.

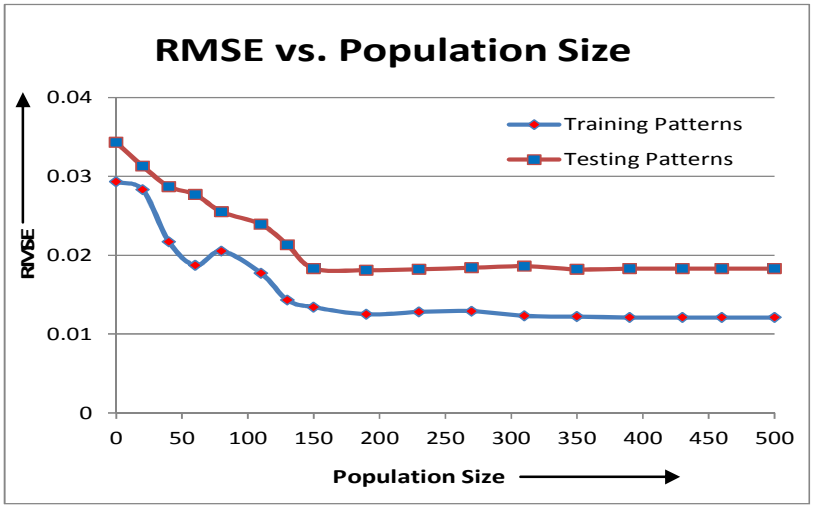


Figure 9.3: RMSE for best fitted population member vs population size

From the analysis, size of population for SFLA which contains set of premise parameters has been finalised as 150 and distribution of population has also been done by following structure of SFLA: 10 memplexes are assumed to be formed and each of which will contain 30 population members. Consecutively, 5 members are predicted to be grouped together to form a sub-memplex. So, 3 sub-memplexes will be present within each memplex.

To verify the authenticity of proposed hybrid learning mechanism, it has been compared here with two most popular hybrid learning techniques such as GD+LSE [77] and PSO+LSE [206]. Values of control parameters for both algorithms have been used here by following previous researches as listed in Table 9.1. At first, mentioned hybrid algorithms have been implemented as learning scheme for developed ANFIS architecture with respect to the same training and testing data sets which has already been used for proposed training method. The performances of all training algorithms have been recorded to perform a fair comparison.

Table 9.2: Control parameters for previously developed hybrid training algorithms of ANFIS

Algorithms	Details of Control Parameters
GD+LSE [77]:	Step size, k can be varied between (0.01 to 0.09), here k=0.01
PSO+LSE[206]:	Cognitive acceleration, c1=2, Social acceleration, c2 =2, Values of r1 and r2 are randomly distributed in [0,1], Initial inertia weight ω_{min}= 0.1 and Final inertia weight ω_{max}= 0.5

Variation in estimated outputs for both ANFIS models has been observed for different training algorithms. A comparative study has been executed between three hybrid learning algorithms in Table 9.3 and 9.4 with respect to the predicted outputs of training and testing data sets respectively which have already given in Chapter 4. The relative errors between predicted and estimated outputs have been calculated for each training algorithm and average of these errors has also been plotted as bar chart in Figure 9.4(a) and 9.4(b) for training and testing patterns respectively. In this analysis, the proposed SFLA-RLS based training algorithm has outperformed the performances of PSO-LSE [206] and GD-LSE [77] based learning methods by providing lowest error between actual and predicted outputs of ANFIS models.

Table 9.3: Comparison between predicted and actual values of Δψ and Δθ for training data set

Training Patterns	Predicted value of Δψ (in degree)	Estimated value of Δψ (in degree)			Predicted value of Δθ (in degree)	Estimated value of Δθ (in degree)		
		SFLA-RLS	PSO-LSE[145]	GD-LSE[128]		SFLA-RLS	PSO-LSE[145]	GD-LSE[128]
(i)	-5	-5.05	-5.13	-4.89	11.45	11.59	11.73	11.87
(ii)	-4.78	-4.71	-4.9	-4.97	4.29	4.17	4.42	4.56
(iii)	12.39	12.32	12.25	12.51	26.35	26.21	26.58	26.67
(iv)	-15.47	-15.53	-15.66	-15.67	-17.23	-17.35	-17.04	-17.59
(v)	3.51	3.49	3.63	3.69	-8.09	-8.46	-8.75	-8.83
(vi)	-14.3	-14.23	-14.39	-14.17	19.47	19.33	19.72	19.87
(vii)	7.5	7.43	7.56	7.62	-13.09	-13.31	-12.77	-13.54
(viii)	10.89	10.81	10.97	11.03	21.14	20.94	21.45	21.63
(ix)	-2.07	-1.97	-2.16	-2.25	7.96	8.05	8.22	8.35
(x)	8.34	8.19	8.59	8.65	-15.75	-15.58	-16.09	-16.26
(xi)	-21	-21.36	-20.34	-21.87	9.83	9.95	9.57	10.18
(xii)	-15.07	-14.89	-15.35	-15.57	3.64	3.74	3.81	3.89
(xiii)	9.41	9.51	9.25	9.72	24.05	23.92	23.78	24.56
(xiv)	14.35	14.47	14.69	14.73	-12.23	-12.39	-12.52	-12.61
(xv)	-10.23	-10.17	-10.45	-10.52	-16.56	-16.25	-16.96	-17.14
(xvi)	-2.59	-2.65	-2.71	-2.82	7.89	8.03	7.64	8.28
(xvii)	5.87	5.93	6.08	6.12	-13.52	-13.35	-13.82	-13.95
(xviii)	-13.09	-12.97	-13.34	-13.57	27.29	27.11	27.58	27.82
(xix)	7.21	7.32	6.99	7.48	-15.77	-15.89	-15.53	-16.11
(xx)	3.56	3.37	3.75	3.86	2.63	2.71	2.87	2.94
(xxi)	4.25	4.13	4.42	4.56	10.95	10.79	11.18	11.37

Due to stochastic nature, optimal solutions of evolutionary algorithms in one simulation run may be different from other runs of the algorithm. To avoid any vagueness or uncertainties in optimal value for ANFIS parameters as evolved by modified SFLA-RLS based learning scheme, training and testing of ANFIS models have been conducted for 20 times. Average of all recorded RMSEs as experienced by each ANFIS model during training and testing period has also been calculated and considered as lowest RMSE for that specific ANFIS model.

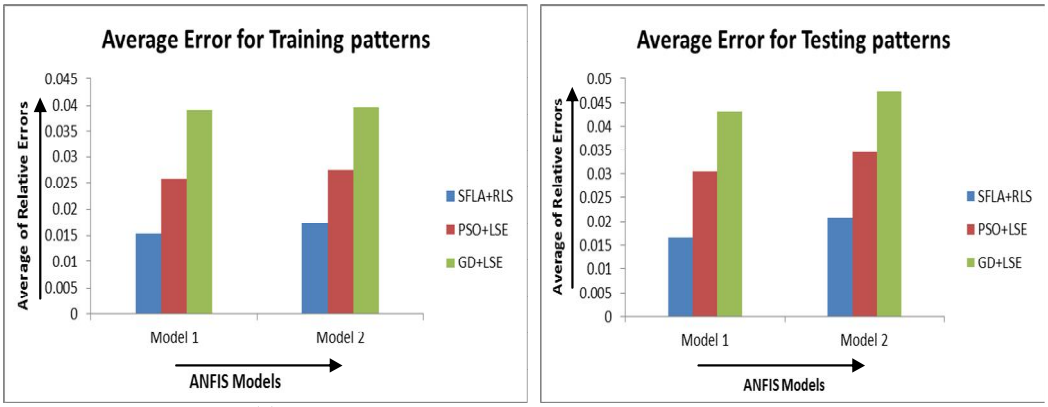


Figure 9.4: Bar chart for average of relative errors of ANFIS models trained by different learning algorithm based on (a) training patterns and (b) testing patterns

Comparison between average performance of learning schemes using same training and testing data sets has been conducted (in Table 9.5) to ensure the reliability of proposed hybrid learning method. It has been observed in that proposed SFLA-RLS combination can provide less RMSE in average for both ANFIS models in comparison with PSO-LSE [206] or GD-LSE [77] based training. The set of the premise and consequent parameters associated with average RMSE has been treated as optimal parameters for ANFIS models.

Table 9.4: Comparison between predicted and actual values of $\Delta\psi$ and $\Delta\theta$ for testing data set

Testing Patterns	Predicted value of $\Delta\psi$ (in degree)	Estimated value of $\Delta\psi$ (in degree)			Predicted value of $\Delta\theta$ (in degree)	Estimated value of $\Delta\theta$ (in degree)		
		SFLA-RLS	PSO-LSE[206]	GD-LSE[77]		SFLA-RLS	PSO-LSE[206]	GD-LSE [77]
(i)	-6.9	-6.74	-7.17	-7.25	16.26	16.43	15.97	16.59
(ii)	-11.47	-11.61	-11.16	-11.83	-9.71	-9.53	-9.96	-10.12
(iii)	15.23	15.02	15.59	15.71	-15.84	-15.97	-15.63	-16.19
(iv)	7.94	7.73	8.23	8.44	-4.59	-4.47	-4.91	-5.03
(v)	-21.56	-21.81	-22.04	-22.18	23.65	23.82	23.99	24.15
(vi)	-13.42	-13.23	-13.69	-14.17	-17.33	-17.19	-17.03	-17.68
(vii)	18.05	18.33	18.56	18.62	14.58	14.87	14.16	15.06
(viii)	5.79	5.61	6.11	6.19	6.39	6.28	6.67	6.75
(ix)	-4.53	-4.69	-4.26	-4.82	5.87	5.73	6.05	6.19

Such negligible errors may ensure that the developed ANFIS architecture trained by proposed SFLA-RLS algorithm possesses enough robustness and efficiency to find out accurate change in steering angle required for safe navigation. The average computation time required to train each ANFIS model by proposed training method has also been noted and compared with previously developed hybrid learning schemes in Table 9.5.

Table 9.5: Comparison in terms of average value of RMSE and computation time

Optimization method	Average RMSE during Training		Average RMSE during Testing		Average Computation time (in sec)	
	ANFIS Model for $\Delta\psi$	ANFIS Model for $\Delta\theta$	ANFIS Model for $\Delta\psi$	ANFIS Model for $\Delta\theta$	ANFIS Model for $\Delta\psi$	ANFIS Model for $\Delta\theta$
GD+LSE [77]	0.033	0.031	0.057	0.081	1.79	1.43
PSO+GD [206]	0.023	0.029	0.035	0.47	0.83	0.97
SFLA+RLS	0.0002	0.005	0.0047	0.0052	0.49	0.61

The number of ANFIS Epochs required for training is also a matter of concern. Average variation in RMSE for each hybrid training algorithm with respect to the epochs has been plotted here in Figure 9.5(a) and 9.5(b) during training and testing periods respectively. As RMSE has been assumed as fitness value of premise parameter set, so, degradation in RMSE is desirable and corresponding curve has been treated as fitness convergence curve. In a generalized manner, 500 epochs are assigned to train the proposed ANFIS topology using three different hybrid learning algorithms separately.

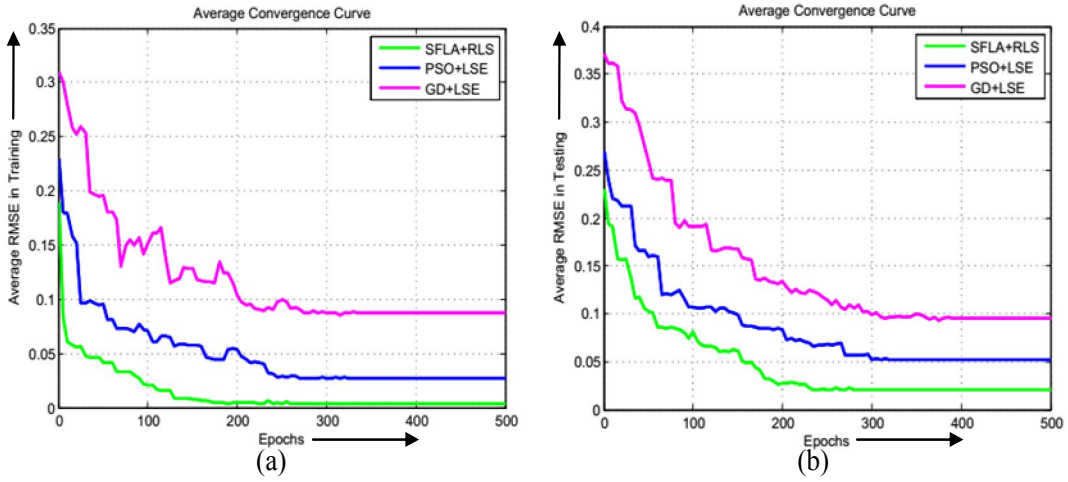


Figure 9.5: Average convergence curve during (a) Training and (b) Testing periods

By observing Figure 9.5, it has been found that considered learning schemes have completed their training within first 300 epochs by stabilizing the root mean square error at very low value. Maximum permissible error has been declared as 0.009.

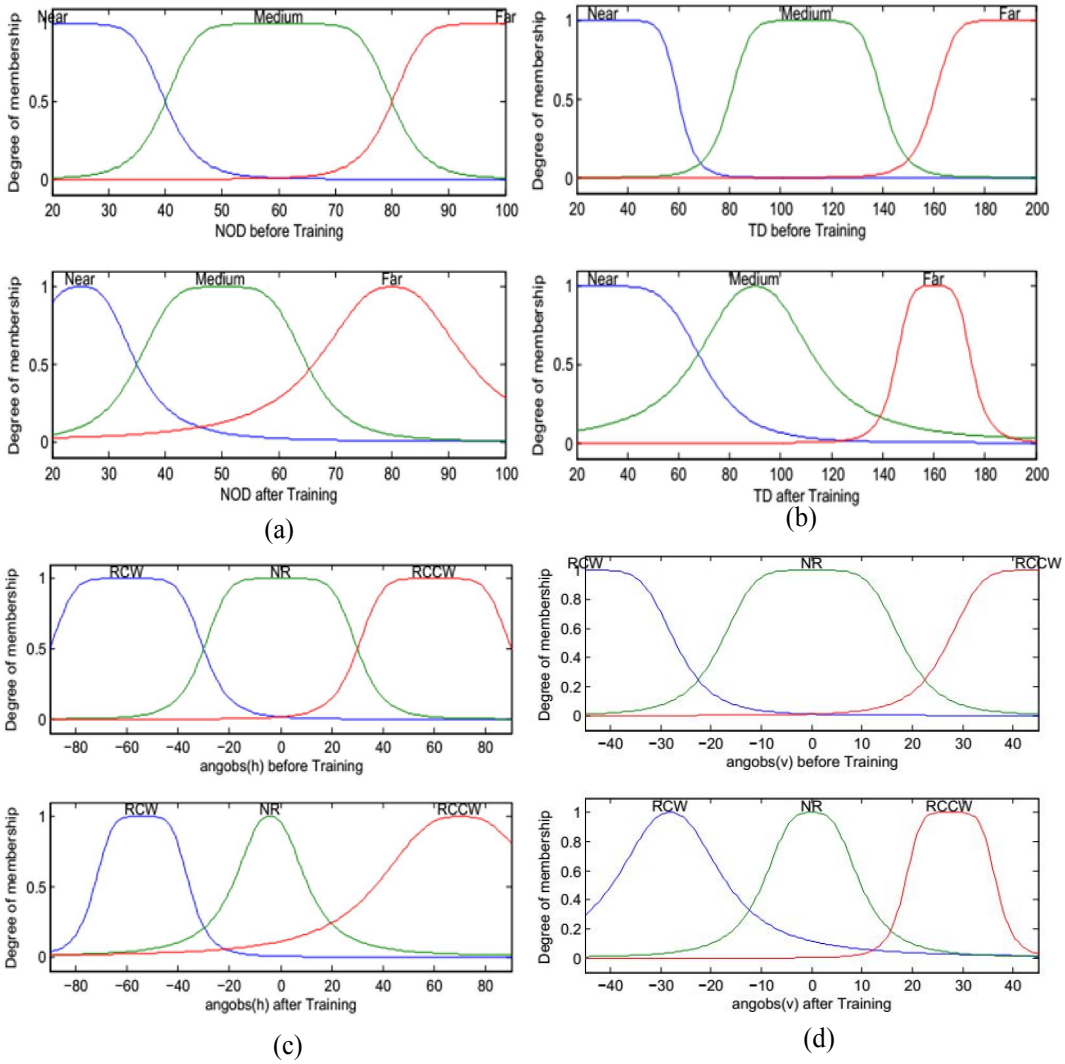


Figure 9.6: Membership Functions for Input Variables before and after training: (a) NOD (Nearest obstacle Distance), (b) TD (Target Distance), (c) angobs(h) (Obstacle angle in horizontal plane), (d) angobs(v) (Obstacle angle in vertical plane)

Note: Changes in parameters and shape of membership functions for angtar(h) and angtar(v) have also computed same as angobs(h) and angobs(v) respectively.

From the comparative assessment of Figure 9.5 (a) and (b), it has been found that the proposed SFLA-RLS based hybrid training scheme can achieve stable value of RMSE more quickly than other two learning schemes in both training and testing modes. Lowest value of RMSE has also been recorded for SFLA-RLS based learning scheme. It denotes a perfect match between actual and predicted output of ANFIS. So, it can be stated that proposed hybridization of evolutionary concept and mathematical estimation in training process of ANFIS can perfectly tune the premise and consequent parameters to minimize RMSE by employing much less run time than ANFIS's other popular learning schemes. Change in shape of membership functions for input variables after training has been viewed in Figure 9.6. Proposed algorithm has been implemented here to avoid local minima situation as well as to enhance convergence speed by maintaining a trade-off between the exploration and exploitation abilities of training program for ANFIS.

9.6 Simulation Results

Faster error minimization capability of proposed ANFIS framework trained with modified SFLA-RLS scheme has inspired the current research work for implementing it as a navigational strategy of underwater robot. On detection of obstacles by sensors, underwater robot must change its direction of motion or heading angle to move towards the specified target without any collision with obstacles. Based on online sensor readings, proposed ANFIS topology has been employed here to estimate the precise change in heading angle of underwater robot required for avoiding obstacles safely. Apart from obstacle avoidance, present research work has made an attempt to minimize the path length travelled by robot during underwater navigation. Details of hypotheses required for simulation of three-dimensional navigation have already been discussed in Section 4.5 of Chapter 4. To execute reactive behaviours (e.g. obstacle avoidance, wall following, target seeking etc.), simulation exercises have been performed for virtual underwater robot in three-dimensional scenario of MATLAB (Figures 9.7-9.9) where obstacles of different shape and sizes are randomly distributed and start and target coordinates are assumed to be known. Underwater robot embedded with designed ANFIS rules can effectively trace collision free near optimal path from start to goal in unknown or partially known environment of simulation mode (Figures 9.7-9.9). Figure 9.8 has shown the ability of proposed navigational strategy to avoid dead end condition of obstacles using wall following and target seeking behaviours.

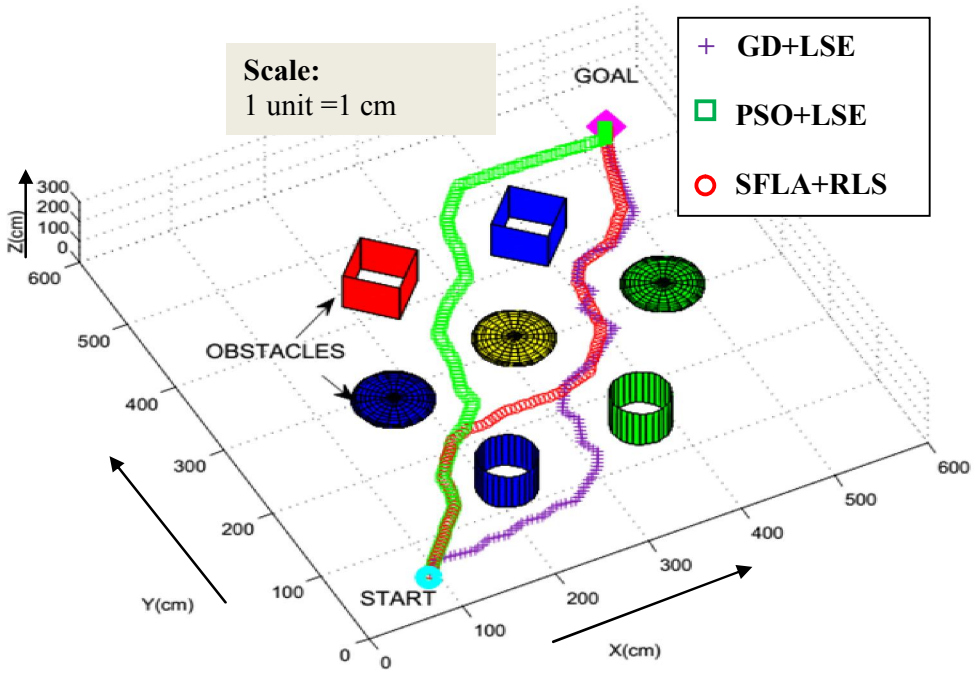


Figure 9.7: Simulation result for obstacle avoidance and target seeking behaviour

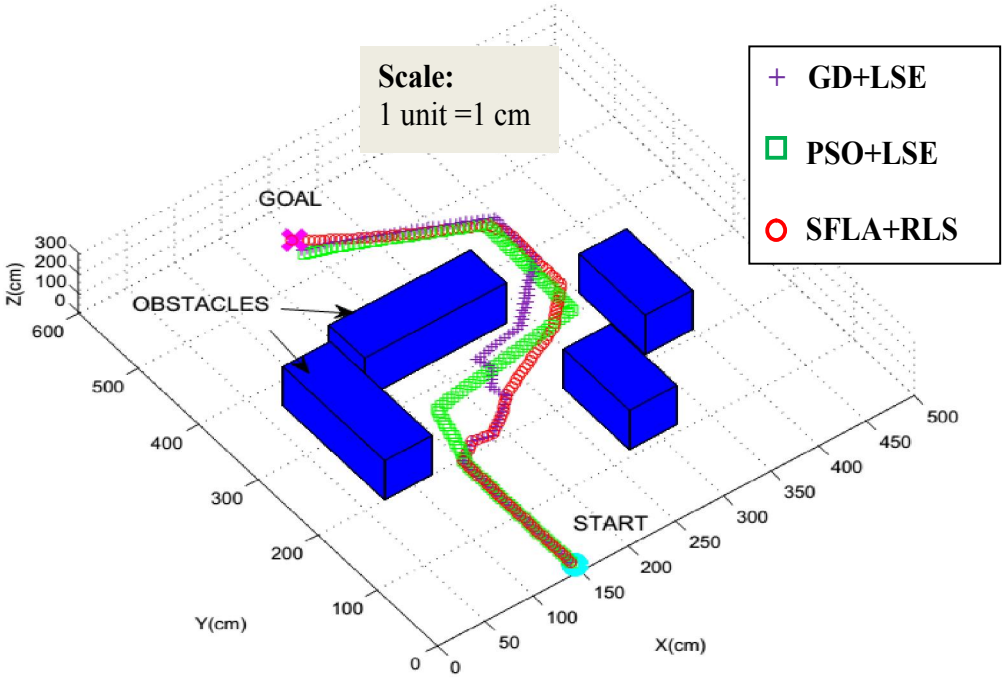


Figure 9.8: Wall following behaviour to escape dead end condition

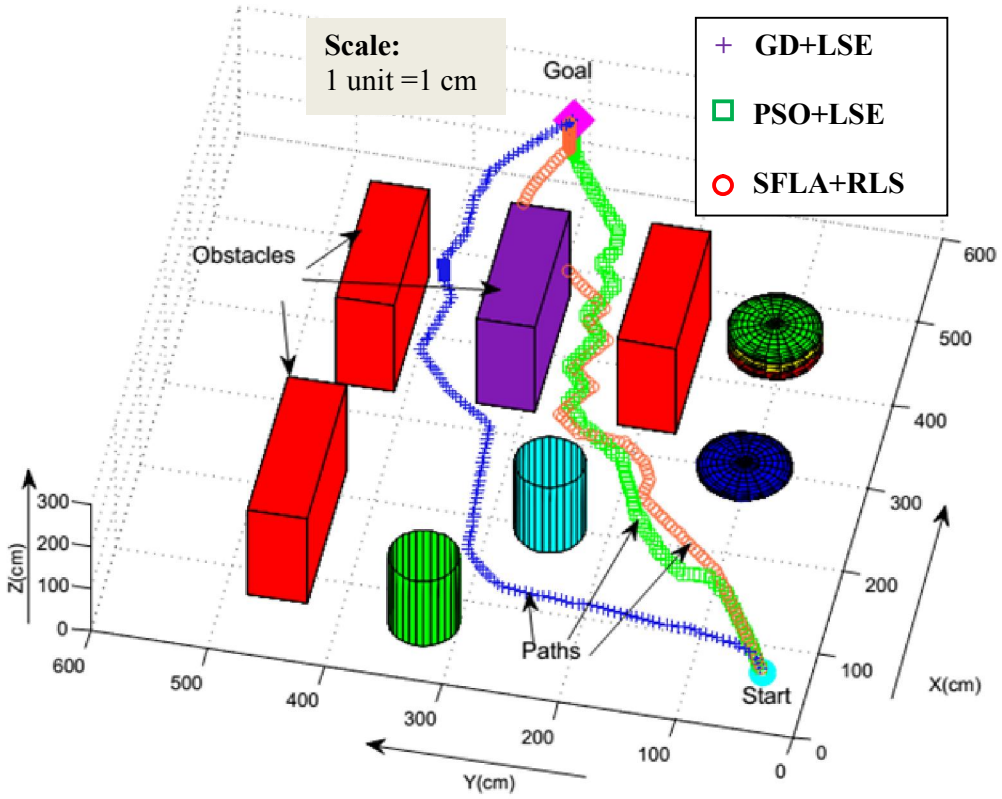


Figure 9.9: Simulation result for tracing collision free path in a cluttered environment

9.6.1 Navigational Performances of ANFIS for different hybrid learning algorithms:

Navigational performance of proposed hybrid training scheme as employed in developed ANFIS topology has been compared with other two hybrid learning methods (PSO+LSE and GD+LSE) in all simulated scenarios (Figure 9.7-9.9). Details of control parameters for all training algorithms have already mentioned in previous sections. The comparative analysis has been accomplished regarding path length and travel time in Table 9.6 and 9.7 respectively. It has been observed that performance of proposed SFLA-RLS based learning algorithm outperforms other two hybrid learning schemes of ANFIS during three-dimensional path planning. Moreover, proposed navigational strategy has obtained smoother path associated with safe clearance from obstacles than other methods in all simulation results (Figure 9.7-9.9). Simulations have been performed for twelve times for each scenario and the most viable paths traced by robot with minimum length have been viewed here.

Table 9.6: Comparative analysis on various training algorithms of ANFIS regarding path length

Simulation Scenarios	Path length (in cm) for different navigational strategies		
	ANFIS trained by Proposed SFLA-RLS scheme	**ANFIS trained by PSO-LSE algorithm**	**ANFIS trained by GD-LSE method**
Figure 9.7	655.2	674.6	687.8
Figure 9.8	565.9	583.9	576.5
Figure 9.9	636.5	654.8	658.7

Table 9.7: Comparative analysis on various training algorithms of ANFIS regarding travel time

Simulation Scenarios	Travel time (in sec) for different navigational strategies		
	ANFIS trained by Proposed SFLA-RLS scheme	**ANFIS trained by PSO-LSE algorithm**	**ANFIS trained by GD-LSE method**
Figure 9.7	18.72	23.26	25.47
Figure 9.8	16.17	20.14	21.35
Figure 9.9	18.19	22.58	24.40

9.6.2 Comparison with Other Navigational Approaches

In this section, the prediction ability of proposed ANFIS architecture trained with SFLA-RLS based hybrid learning approach has been verified with respect to the performances of fuzzy logic based approach [156] and dynamic 3D model of neural network [207] during three-dimensional navigation. For comparative study, proposed ANFIS architecture trained with SFLA-RLS based hybrid learning approach has been implemented in almost similar simulation scenarios as used by the previous researchers. The simulation results and table for comparison have been narrated as follows:

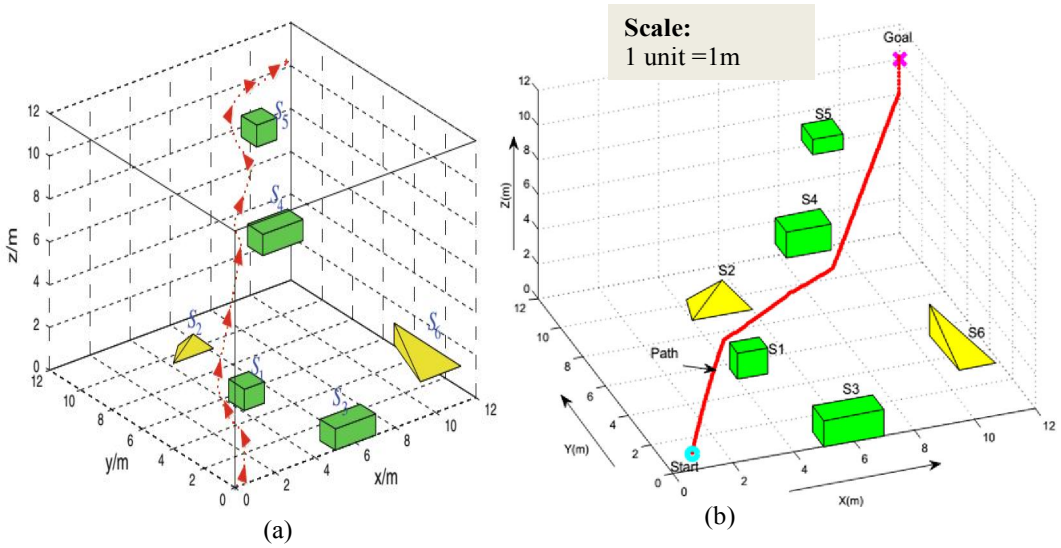


Figure 9.10: (a) Three-dimensional static path planning for AUV using fuzzy logic control by Jiang and Zhu [187]; (b) Navigational path traced by proposed ANFIS topology trained with SFLA-RLS based hybrid learning algorithm

(a) In Chapters 6 and 7, Navigation of AUV based on Fuzzy logic controller as proposed by Jiang and Zhu [187] has already been considered for executing comparison of navigational performances. The same scenario (Figure 9.10 (a)) has been considered here to implement Proposed ANFIS based navigational strategy. Figure 9.10(b) has shown the simulated path traced by recent algorithm. Comparison between two approaches regarding path length has been exhibited in Table 9.8. Proposed navigational strategy based on ANFIS topology trained with SFLA-RLS scheme has traced shorter path than Fuzzy logic controller of Jiang and Zhu [187]. Successful avoidance of collision has also been observed during navigation.

(b) Yan et al. [207] have proposed path planning strategy for AUV based on neuro-dynamics of three-dimensional neural network. Workspace of underwater robot has been discretised into subspaces to represent neurons of neural network's 3D dynamic model. Neurons related to target region are assumed to be excited by high value of neural activity which are globally propagated within connections of network. On the other hand, from the obstacle regions, neural network has received negative value locally which helped to avoid obstacles. Without any training or any prior map of workspace, Yan et al.[207] have planned path for AUV within the simulated environment of 10x10x10 grid map

which contains four static obstacles in between starting position (1, 2, 1), and target position (10, 10, 10) as shown in Figure 9.11 (a). The proposed ANFIS architecture trained by hybrid of SFLA and RLS schemes has been applied as navigational strategy in simulated environment of Figure 9.11 (b) which has been developed here to replicate the scenario given in Figure 9.11 (a). Less path length has been recorded for proposed navigation scheme than previous method developed by Yan et al. [208] in Table 9.8.

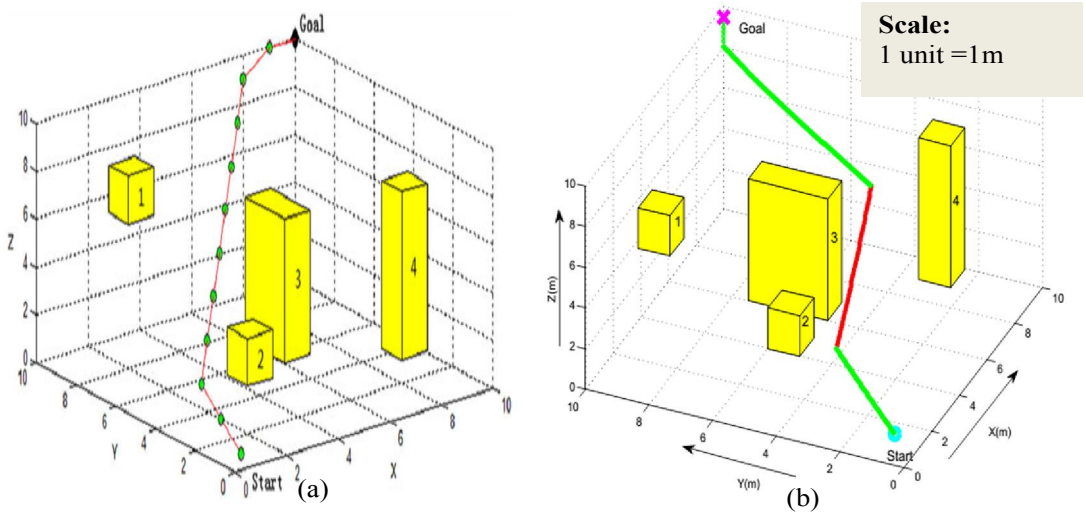


Figure 9.11: (a) Simulation result for fully connected 3D model of neural network by Yan et al. [207]; (b) Navigational path traced by proposed ANFIS topology trained with SFLA-RLS based hybrid learning algorithm

Table 9.8: Numerical Comparison between Proposed ANFIS trained with SFLA-RLS based hybrid learning approach and other navigational strategies such as advanced version of Fuzzy based Controller and Neural Network Model

Figure No.	Navigational Strategy	Length of path traced by robot (in cm)	Minimization (in %)
9.10(a)	Fuzzy Logic Controller [187]	223.9	6.13
9.10(b)	ANFIS trained with SFLA-RLS learning scheme	210.7	
9.11(a)	3-D Neural Network Model [207]	796.4	3.2
9.11(b)	ANFIS trained with SFLA-RLS	771.6	

Smoothness of path has also been increased in Figure 9.11 (b) than Figure 9.11 (a). Moreover, 3D dynamic model of neural network may not be successful for complex situations like dead end conditions as it does not have any learning facilities. In this

context, proposed navigational strategy has found to be more profitable than the previous one.

9.7 Experimental Results and Comparisons

Navigational strategy based on proposed ANFIS topology trained by hybrid learning mechanism has been successfully implemented within navigational controller of underwater robot "GNOM-Baby". Verification of simulation results has been successfully executed in real-time underwater environment where obstacles are positioned by seeing the simulation scenario of Figure 9.9. Figure 9.12 has shown the experimental views of underwater robot "GNOM-Baby" at different stages of navigation. It has been observed that underwater robot has successfully avoided obstacles of different shape and sizes by keeping safe distance from them while moving from start to goal point in real time underwater environment. Comparison between experimental and simulation results has been carried out here regarding path length and travel time taken by underwater robot during navigation.

9. 7.1 Comparison between simulation and experimental results

Due to stochastic nature of SFLA, its population set will be generated in a random manner as seen in Chapter 5. Solution vectors or premise parameter sets in population may not be same in two consecutive training of proposed ANFIS architecture. So, global best values of antecedent parameter set may differ in different runs of algorithm. Therefore, simulation study (Figure 9.9) and its corresponding experimental verification (Figure 9.12) have been repeatedly performed for 20 times in same environment to ensure the performance of proposed Adaptive SFLA based training scheme. Variation in premise parameters directly affects the ANFIS models' outputs which changes the turning angle of underwater robot in each run. Therefore, different path length and travel time has been recorded in Table 9.8 and Table 9.9 for each run.

The difference between simulated and experimental performances in terms of path length has been computed in Tale 9.8. Deviations in the length of actual paths with respect to optimum path, which is a straight distance between start and goal points of environment, have also been observed. The shortest distance between start and goal positions in given scenario has been recorded as 607.5 cm and 643.9 cm for simulation (Figure 9.9) and experimental (Figure 9.12) scenarios respectively.

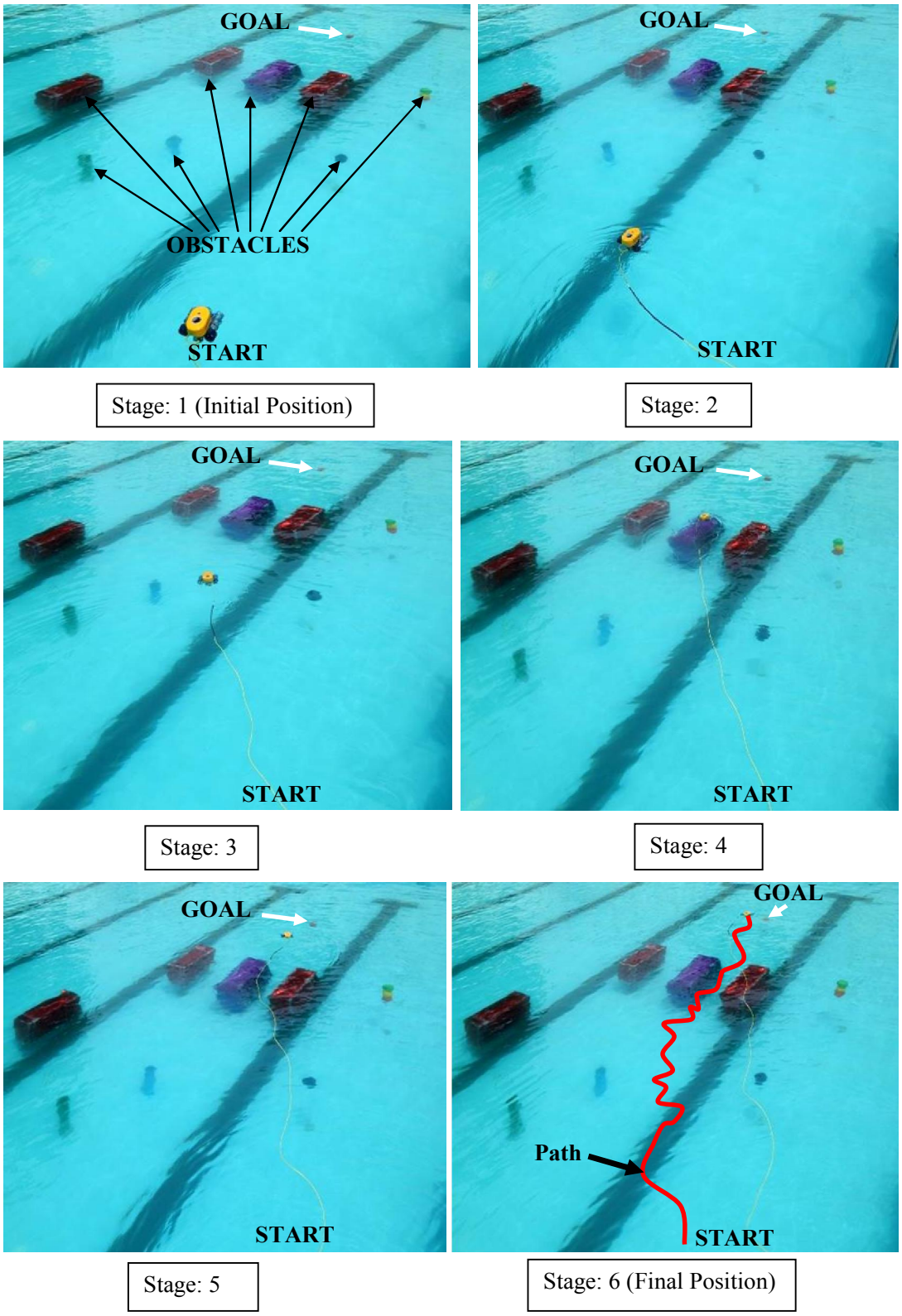


Figure 9.12: Experimental view for underwater navigation of GNOM Baby embedded with ANFIS trained with SFLA-RLS based hybrid learning scheme

For both simulated and real time environments, paths with minimum length have been considered as best paths which have been shown in Figure 9.9 and 9.12 respectively. Shortest path has obtained in simulation mode at 10^{th} run and in experimental mode at 15^{th} run (Table 9.8).

Table 9.9 Comparison between experimental and simulated results regarding path length for Scenario1 (Figure 9.9 and 9.12)

1	2	3	4	5	6
No. of Runs	**Path length in simulated mode (in cm)**	**Deviation from Optimum path (%)**	**Path length in experimental mode (cm)**	**Deviation from Optimum path (%)**	**Difference between column 2 and 4 (%)**
1^{st}	634	4.36	669.1	3.76	5.53
2^{nd}	639.4	5.24	673.8	4.43	5.38
3^{rd}	637.2	4.87	670.8	4.01	5.29
4^{th}	636.7	4.81	667.9	3.59	4.89
5^{th}	631.3	3.92	665.8	3.28	5.45
6^{th}	632.4	4.11	662.7	2.84	4.79
7^{th}	635.3	4.56	663.6	2.97	4.47
8^{th}	634.8	4.50	665.1	3.18	4.76
9^{th}	636.4	4.76	670.2	3.92	5.31
10^{th}	**627.3**	**3.27**	663.7	2.98	5.79
11^{th}	637.6	4.95	670.3	3.94	5.13
12^{th}	636.2	4.71	669.4	3.80	5.22
13^{th}	633.3	4.25	663.2	2.91	4.72
14^{th}	635.2	4.56	665.6	3.26	4.78
15^{th}	631.2	3.8	**662.5**	**2.81**	4.97
16^{th}	629.3	3.59	664.7	3.12	5.62
17^{th}	632.1	4.04	662.8	2.85	4.87
18^{th}	631.3	3.92	664.1	3.04	5.19
19^{th}	634.1	4.36	664.2	3.05	4.76
20^{th}	636.4	4.75	669.7	3.85	5.23
Average difference in path length between simulated and experimental results					5.11

Simulations of three-dimensional navigation and their corresponding experimental verifications have been conducted for a number of scenarios by changing the positions of start point, goal point and obstacles. Observations on executed investigations in terms of in terms of path length and travel time have been noted in in Table 9.10 and Table 9.11 respectively. Navigational performance of proposed ANFIS structure trained with SFLA-

RLS based hybrid learning strategy have been found to be quite satisfactory in all simulated and experimental scenarios.

Table 9.10 Comparison between experimental and simulated results for travel time in Scenario1 (Figure 9.9 and 9.12)

1	2	3	4
No. of Runs	**Travel time in simulated mode (in sec)**	**Travel time in experimental mode (in sec)**	**Difference between column 2 and 4 (in %)**
1st	10.39	10.84	4.33
2nd	10.69	11.10	3.82
3rd	10.45	10.94	4.77
4th	12.49	13.12	5.10
5th	9.87	10.47	6.11
6th	12.91	13.46	4.30
7th	11.55	12.09	4.66
8th	8.70	9.12	4.90
9th	12.24	12.84	4.91
10th	7.05	7.43	5.43
11th	8.28	8.78	6.00
12th	7.15	7.55	5.58
13th	8.33	8.69	4.31
14th	7.30	7.62	4.42
15th	7.99	8.37	4.80
16th	7.15	7.54	5.46
17th	9.43	10.02	6.25
18th	8.20	8.61	5.01
19th	10.23	10.69	4.54
20th	11.17	11.72	4.99
Average difference in travel time between simulated and experimental results			4.98

Table 9.11 Comparisons between experimental and simulated results for five more scenarios regarding path length

Different Scenarios	Best path length in Simulation (in cm)	Best path length during Experiment (in cm)	Difference (in %)
Scenario-2	655.3	691.1	5.46
Scenario-3	971.5	1021.5	5.14
Scenario-4	725.4	762.6	5.12
Scenario-5	1027.4	1081.4	5.25
Scenario-6	873.8	917.3	4.98

Table 9.12 Comparisons between experimental and simulated results for five more scenarios regarding travel time

Different Scenarios	Minimum travel time during simulation (in sec)	Minimum travel time during experiment (in sec)	Difference (in %)
Scenario-2	17.71	18.60	5.00
Scenario-3	28.57	30.03	5.11
Scenario-4	19.61	20.56	4.84
Scenario-5	29.36	30.94	5.40
Scenario-6	24.97	26.14	4.68

9.8 Summary

In this chapter, ANFIS topology trained with SFLA-RLS based hybrid learning algorithm has been explored as an effective path optimization technique for underwater robot. To validate the proposed navigational approach, numerous simulation and experimental results have been performed and analysis on navigational performance has been précised as follows:

- ANFIS's learning facility and human like ability to deal with ambiguities or vagueness present in the environment have been combined to estimate change in heading angle for any pose of underwater robot during online navigation.

- Two major objectives of path planning problem, obstacle avoidance and minimization of path length have been achieved here by regulating the turning angles of underwater robot in both horizontal and vertical planes of chaotic 3D workspace.

- A trade-off between computational complexity and exactness in training process has been maintained by using population based approach like Adaptive SFLA to train the premise parameters and RLS to estimate consequent parameters of ANFIS model.

- Proposed self-learning mechanism of ANFIS has found to be more reasonable than other hybrid learning mechanisms of ANFIS while performing robotic behaviors within underwater workspace in spite of uncertainties and nonlinearities associated with it.

- Superiority of proposed approach in comparison with other navigational methods (fuzzy logic and 3D dynamic model of neural network) during three-dimensional navigation also validates the feasibility of current algorithm.

- Navigational performance of proposed approach within complex scenarios has been evaluated in simulation and experimental mode. Minor variances between simulated and experimental performances have been observed in average such as 5.18% for path length and 5% for travel time.

- Collision-free near optimal paths traced by underwater robot in both simulated as well as experimental modes ascertain the effectiveness and sturdiness of proposed navigational strategy for underwater motion.

In this chapter, evolutionary based stochastic computation and classical estimation techniques have been coupled together to incorporate some certainty and perfection within human perception based decision making abilities of ANFIS model in an automatic manner. Minimization of prediction error has been considered as major objective of the proposed hybrid learning approach for ANFIS model. Direction of underwater robot's motion has been accurately determined by employing the proposed ANFIS architecture associated with self-learning ability. A comparative study on navigational performances of all proposed three-dimensional path planning algorithms have been executed in next chapter to find out the most efficient among them.

10. Results and Discussions

This chapter has been commended to encapsulate the comparison between navigational performances of all proposed navigational strategies for endorsement purpose. Different benefits as well as drawbacks of human perception based estimation and evolutionary based optimization methods have been analyzed while tracing collision free near optimal path within unknown or partially known three-dimensional simulated or real world scenarios. A trade-off between computational complexity and optimization ability of any path planning algorithm will always be desirable while performing path optimization and collision avoidance for underwater robot. In this dissertation, all proposed navigational strategies have been motivated to combine the features of both local and global search approaches for improving the degree of optimality as well as convergence speed in path optimization process. Less mathematical complexity and easy interpretation are also considered as major aspects while choosing navigational strategies for underwater robot.

10.1 Comparison between Proposed Approaches for Underwater Navigation

In previous chapters, navigational approaches with the different aspects of computational intelligence have been developed to explore the optimum and safe path from source to goal in chaotic 3D workspace. In order to avoid the difficulties inherited by the traditional computations based path planning or trajectory tracking process, Fuzzy logic based reasoning process associated with the self-learning ability of Neural Network and different nature-inspired algorithms have been employed as navigational strategies for underwater robot. Fuzzy systems have the capabilities to deal with the uncertainty or vagueness based on the expertise knowledge while solving the complex problems which are difficult to model due to the nonlinearities. Moreover, parallel learning abilities of Neural Network based on training data sets may enhance the level of certainty in Fuzzy based reasoning process. In recent research work, multiple Adaptive Neuro fuzzy systems have been combined to estimate the accurate heading angle of underwater robot during reactive navigation. Reasonable navigational performance has been observed.

On the other hand, the evolutionary algorithms have emerged as robust techniques for many complex optimization, identification, learning and adaptation problems. Stochastic optimization approaches of less computational complexities such as SFLA, DE and HS

have been successfully implemented as navigational strategies of underwater robot. The mentioned natural behaviour based metaheuristics have been modified to perfectly balance the exploration and exploitation abilities of search process. In present chapter, all proposed navigational strategies have been employed for same simulation as well as experimental scenarios to perform a fair comparison. Figures 10.1-10.6 have shown the performances of proposed navigational approaches. Preliminary robotic behaviors (i.e. obstacle avoidance, target seeking, wall following etc.) have been successfully executed by each newly proposed path planning algorithm during three-dimensional navigation within both simulated and experimental scenarios.

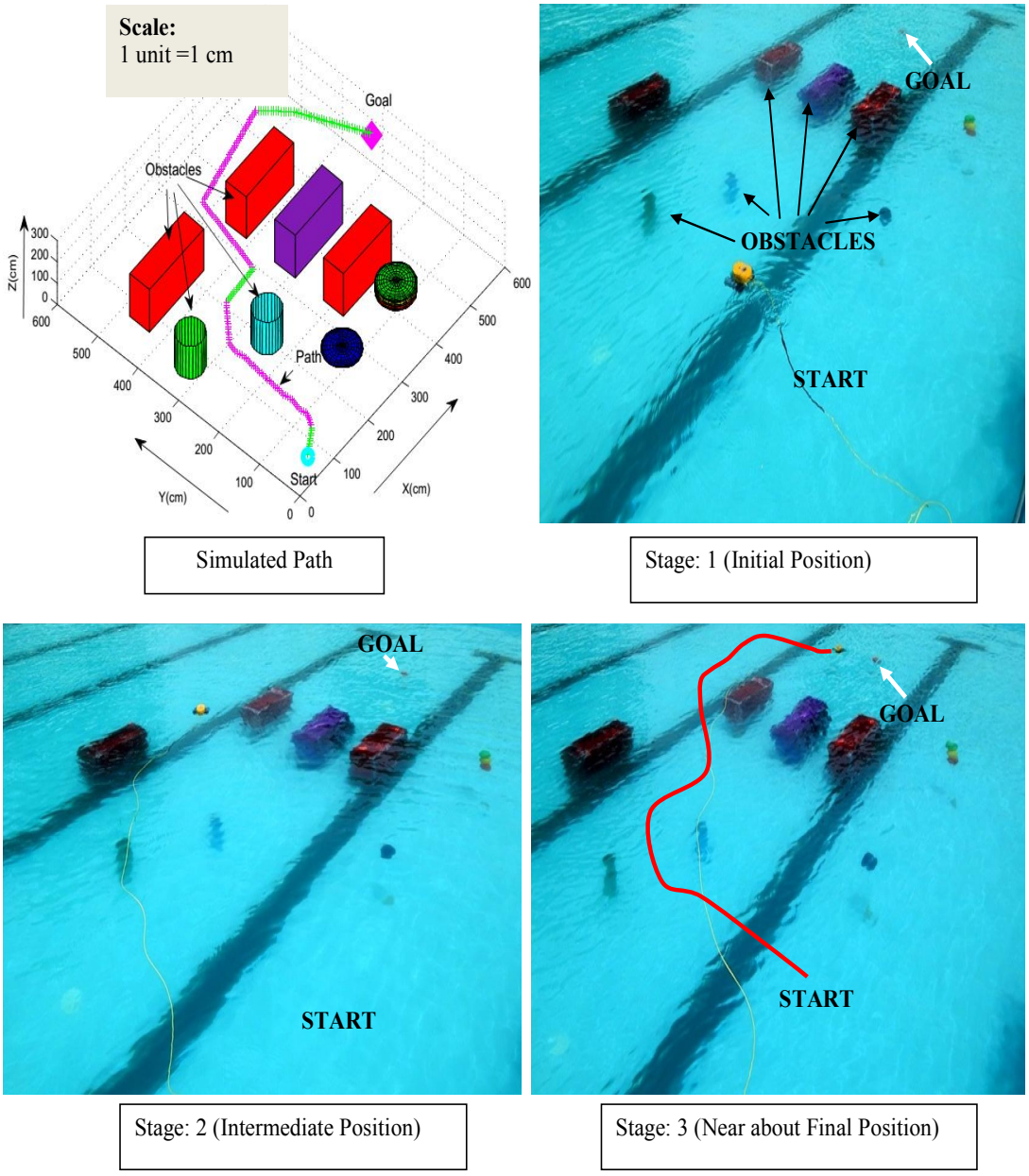


Figure 10.1: Simulation and Experimental Path for underwater robot embedded with Manifold ANFIS based navigational strategy in Scenario1

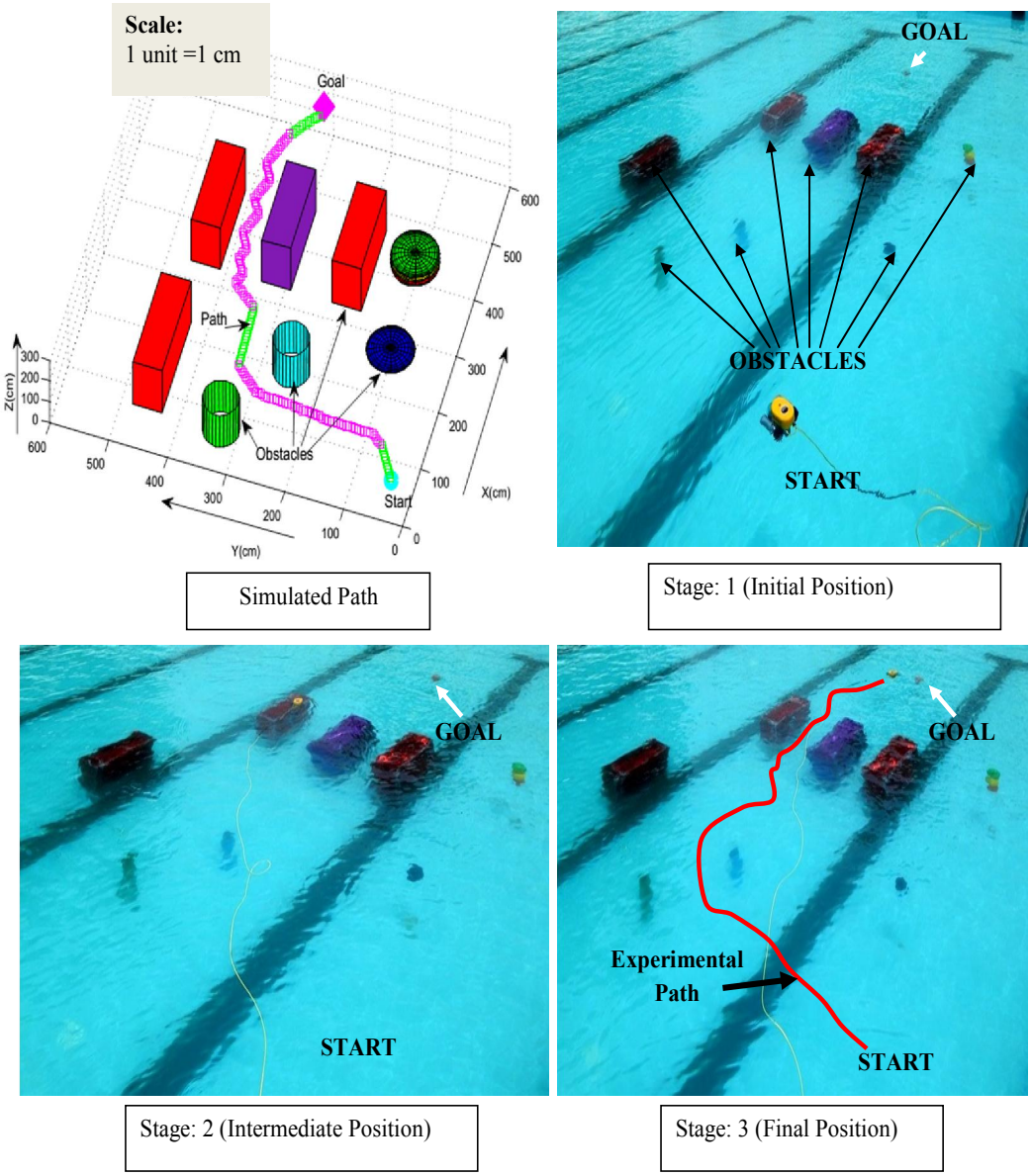


Figure 10.2: Simulation and Experimental Path for underwater robot embedded with Adaptive SFLA based navigational strategy in Scenario1

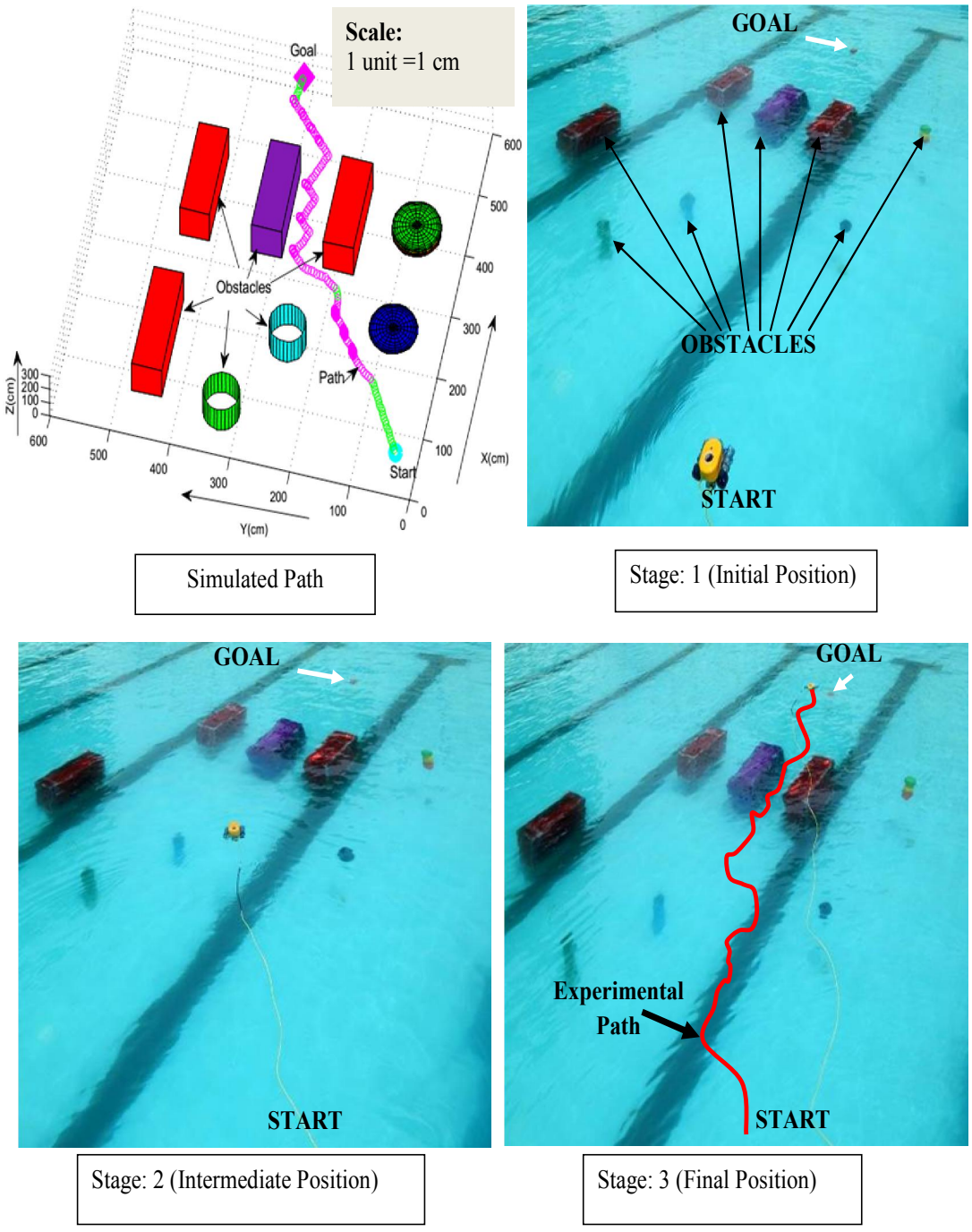


Figure 10.3: Simulation and Experimental Path for underwater robot embedded with Dynamic DE based navigational strategy in Scenario1

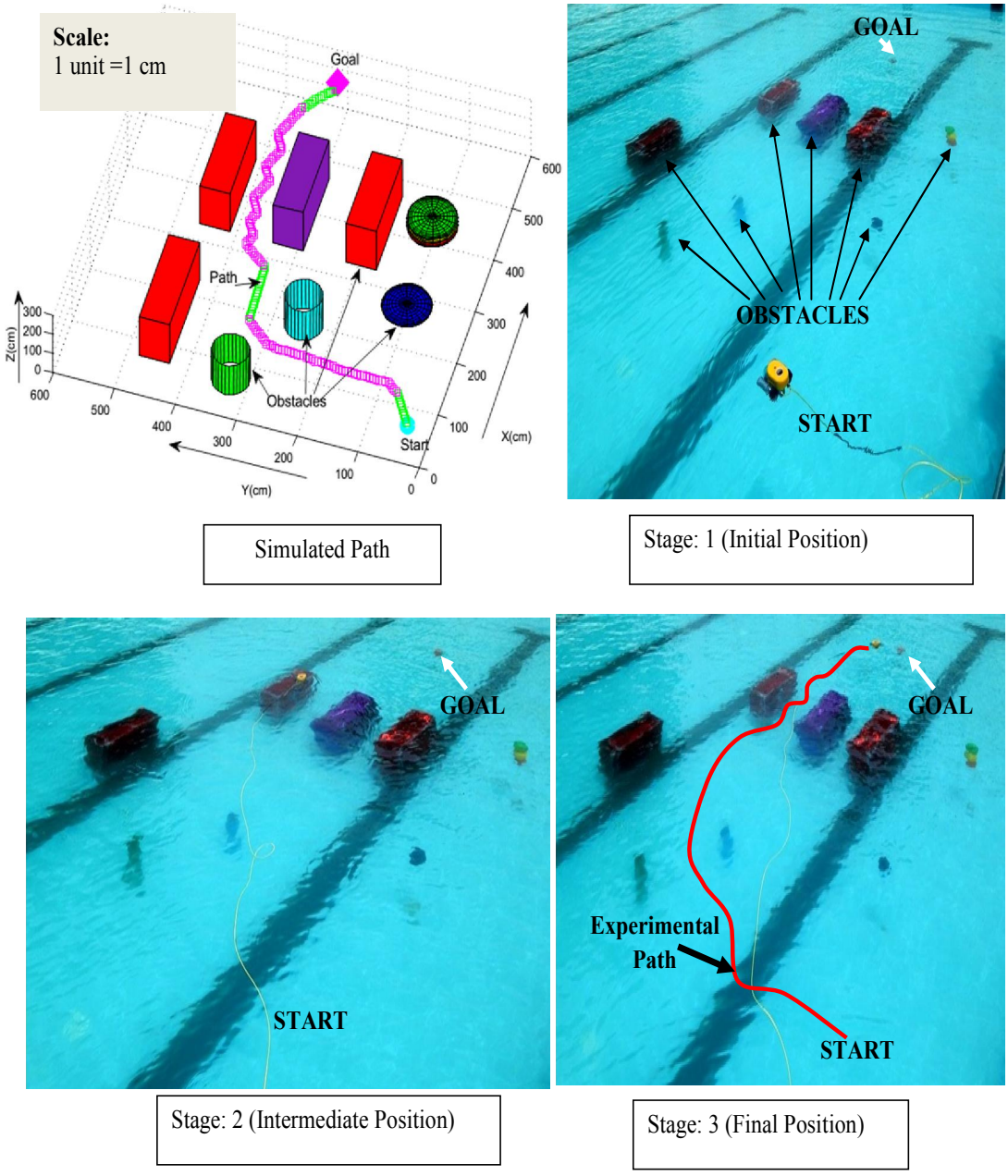


Figure 10.4: Simulation and Experimental Path for underwater robot embedded with Adaptively Tuned Harmony Search based navigational strategy in Scenario1

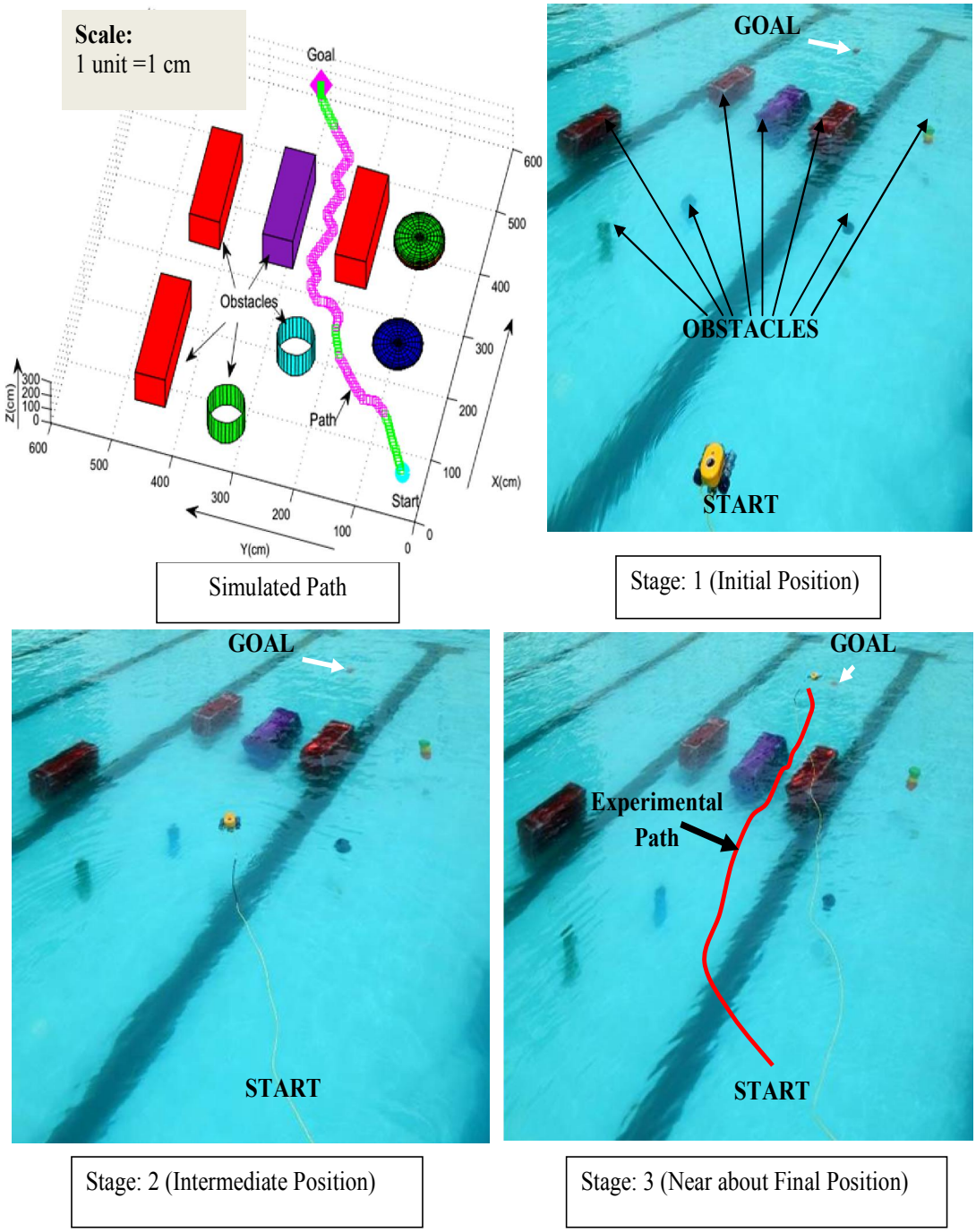


Figure 10.5: Simulation and Experimental Path for underwater robot embedded with Hybrid DE-HS based navigational strategy in Scenario1

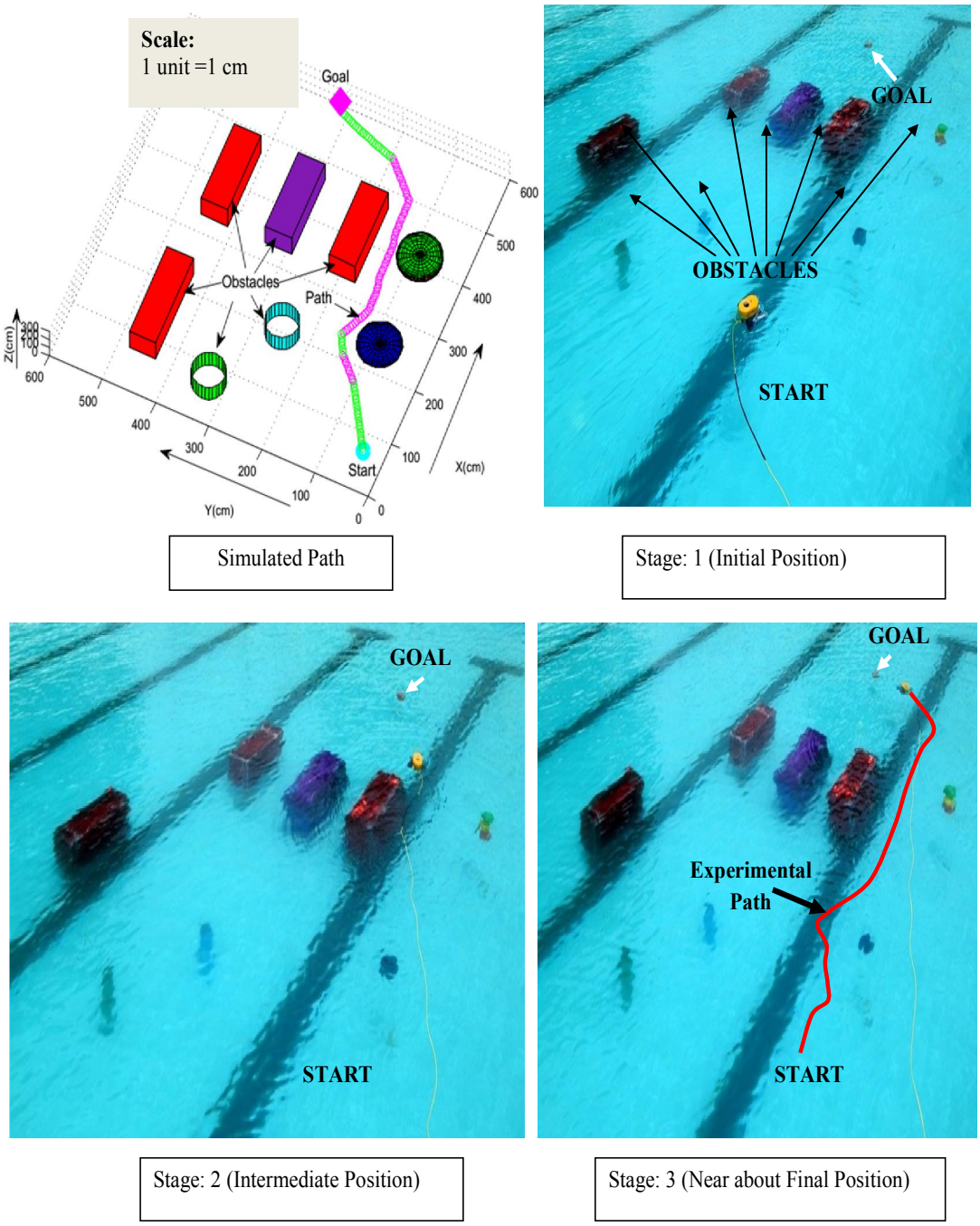


Figure 10.6: Simulation and Experimental Path for underwater robot using Navigational Strategy based on ANFIS model trained with SFLA-RLS based hybrid learning approach in Scenario1

Table 10.1: Comparison between simulation and experimental results for Scenario 1

Navigational Strategies	Average Path length (in cm)			Average Travel Time (in sec)		
	Simulation Result	**Experimental Result**	**Error (in %)**	**Simulation Result**	**Experimental Result**	**Error (in %)**
MANFIS (Figure 10.1)	647.1	687.4	6.23	11.35	12.12	6.79
Adaptive SFLA (Figure 10.2)	639.4	674.9	5.51	10.80	11.41	5.69
Dynamic DE (Figure 10.3)	641.2	678.3	5.79	10.55	11.17	5.96
Adaptively Tuned HS (Figure 10.4)	636.7	670.8	5.35	9.80	10.34	5.51
Hybrid of DE-HS (Figure 10.5)	631.4	663.9	5.15	9.02	9.50	5.35
ANFIS trained with SFLA (Figure 10.6)	629.5	660.8	4.99	8.74	9.19	5.17

The results obtained by implementing various approaches for scenario1 have been recorded in Table 10.1. It shows the percentage of deviation between the simulation and experimental results for various navigational controllers which have been used for finding the navigational path and time taken to reach the target by underwater robot. Table 10.2 has shown comparative study based on more scenarios that have been executed in both simulation and experimental modes. All experiments and simulations have been performed for 20 times and shortest path followed by underwater robot among all trials has been considered as best path and corresponding travel time has also been considered for comparison purpose. Such comparative study has been executed to find out suitable navigational strategy which can trace near optimal collision-free path for underwater robot. Minimization of errors between simulated and experimental performances of considered navigational strategies has also been considered as major research objective to authenticate the feasibility and robustness of proposed navigational approaches.

Table 10.2: Comparison between simulation and experimental results for different scenarios

Scenarios	Navigational Strategies	Average Path length (in cm)			Average Travel Time (in sec)		
		Simulation Result	Experimental Result	Error (in %)	Simulation Result	Experimental Result	Error (in %)
2	MANFIS	729	774.2	6.19	12.79	13.65	6.75
	ASFLA	714.6	753.6	5.47	12.60	13.29	5.47
	DDE	693.3	732.4	5.63	12.21	12.92	5.82
	ATHS	721.9	759.8	5.26	12.73	13.40	5.26
	Hybrid DE-HS	753.2	791.4	5.07	13.27	13.96	5.18
	ANFIS trained with SFLA-RLS	705.3	740.1	4.93	12.44	13.05	4.89
3	MANFIS	1045.3	1108.5	6.05	18.34	19.55	6.61
	ASFLA	732.5	772.5	5.45	12.92	13.62	5.45
	DDE	861.9	911.1	5.71	15.19	16.07	5.82
	ATHS	927.5	976	5.23	16.35	17.21	5.29
	Hybrid DE-HS	758.3	797	5.11	13.38	14.06	5.07
	ANFIS trained with SFLA-RLS	751.7	789.4	5.02	13.26	13.92	4.98
4	MANFIS	931.7	988.9	6.14	16.35	17.44	6.7
	ASFLA	819.3	865.1	5.59	14.45	15.26	5.59
	DDE	567.5	600.3	5.77	10.00	10.59	5.88
	ATHS	742.4	781.8	5.31	13.09	13.79	5.37
	Hybrid DE-HS	882.5	927	5.04	15.57	16.35	5
	ANFIS trained with SFLA-RLS	653.8	686.2	4.96	11.53	12.10	4.92
5	MANFIS	547.5	579.7	5.89	9.6	10.22	6.45
	ASFLA	782.9	825.3	5.41	13.81	14.55	5.41
	DDE	637.4	673	5.59	11.23	11.87	5.7
	ATHS	815.3	857.9	5.23	14.38	15.13	5.25
	Hybrid DE-HS	749.6	787.8	5.1	13.23	13.89	5.06
	ANFIS trained with SFLA-RLS	837.7	879.8	5.02	14.78	15.52	4.96

10.2 Summary

In this present chapter, performances of proposed navigational strategies such as Manifold ANFIS approach, Adaptive SFLA, Dynamic DE, Adaptively Tuned HS, Hybrid of DE-HS and ANFIS trained with hybrid learning based on SFLA-RLS have been evaluated for navigation of underwater robot within various simulation and experimental scenarios embedded with different obstacle arrangements. One example can be viewed in Figures 10.1-10.6. All approaches have shown adequate skills while avoiding obstacles in a safe manner. In average, percentage of errors between simulated and experimental performances regarding path length and travel time has been computed for all mentioned navigational strategies.

In detail investigation, it has been observed that Manifold ANFIS approach based path planner may be effectively implemented along with 6-7% average error between its simulation and experimental performances. Average percentage errors for Adaptive SFLA, Dynamic DE, Adaptively Tuned HS and Hybrid of DE-HS based algorithms have been recorded within 5-6% regarding both path length and time taken. Proposed Adaptive SFLA has shown slightly better performance than Dynamic DE based navigational strategy in terms of average error. But Adaptively Tuned HS based algorithm (5.28% for path length and 5.34 for travel time) has shown more improved performance than both Adaptive SFLA (5.49% for path length and 5.52 for travel time) and Dynamic DE (5.7% for path length and 5.84 for travel time) based navigational strategies by reducing the percentage of error in average. Hybridization of two approaches has been found to be very much useful for improving navigational performance. For Hybrid of DE-HS based path planning approach, average errors of 5.06% and 5.13% have been counted for path length and travel time respectively. ANFIS trained with hybrid learning based on SFLA-RLS scheme has shown more reduction in average error (4.88% for path length and 4.98% for travel time). From executed investigations, it can be concluded that near about optimum performances have been found for two approaches, hybrid DE-HS embedded with adaptive control parameters and ANFIS trained with SFLA-RLS based hybrid learning algorithm during navigation within partially known or unknown underwater environment.

11. Conclusions and Future Direction of Research

The research work carried out in this dissertation has accentuated on suitable navigational strategy for underwater motion which can successfully achieve two major objectives of navigation: obstacle avoidance and path optimization. This chapter may provide a brief overview on the main contributions of executed investigations and ideas for further extension of research work.

11.1 Contributions of Dissertation

A brief view on the novel aspects of this dissertation can be notified as follows:

- Assuming no motion in roll and pitch directions of underwater movement, kinematic and dynamic model has been simplified for small size underwater robot which has been considered here for executing experiments in real time underwater environment.
- Stability issues of considered underwater robot model have been analysed based on Lyapunov's direct method considering certain real time assumptions.
- A new concept of integrating manifold ANFIS models has been found to be beneficial for achieving reactive robotic behaviours of underwater robot during navigation.
- Apart from fuzzy-neural approach, nature inspired evolutionary algorithms of less mathematical complexity have been chosen to optimize three-dimensional path of underwater robot. Adaptive tuning mechanisms have been induced in stochastic metaheuristic approaches such as SFLA, DE and HS to maintain a proper trade-off between exploration and exploitation abilities of search process. Fitness values of population members and current iteration numbers have been used in adaption of control parameters for a specific algorithm. New versions of SFLA, DE and HS have been successfully implemented as three-dimensional navigational strategies to achieve the near optimal safe path during underwater navigation. On detection of obstacle by on-board sensors, underwater robot has been stimulated to follow the next global best positions as chosen by any one of proposed population based approaches (Adaptive SFLA, Dynamic DE and Adaptively Tuned HS) in a sequential manner. Simulation and experimental results have shown the success of recent implementations along with average error of 5-6% which is quite reasonable.

- ❖ Hybridization of two metaheuristic approaches (DE-HS) has been proposed for improving path optimization ability of navigational strategy. The feasibility of proposed DE-HS algorithm has been verified to trace in collision free near optimal path for underwater robot in numerous simulated and experimental scenarios. For such hybrid approach, the percentage of error between simulation and experimental results has been reduced in average.
- ❖ A trade-off between computational complexity and exactness in training process of ANFIS has been successfully introduced by using population based approach like Adaptive SFLA to train the premise parameters and RLS to estimate consequent parameters. Such novel hybrid learning approach has been found to be more precise than other available training algorithms of ANFIS.

11.2 Conclusions

In this research scheme, the challenge has been taken to solve a problem related to navigational path analysis of underwater robots in numerous inconsistent environments. From the performed investigations of this thesis, the findings of each chapter can be highlighted as follows:

- Kinematic and dynamic modelling of conventional underwater robot has been studied in chapter three. Roll and pitch motions are not required to be considered for small size underwater robot of current research work. Therefore, a simplified version of underwater robot's motion equations has been derived and its stability has also been analysed using Lyapunov's candidate function.

- In chapter four, navigational controller has been designed by combining reactive behavioral modules which are composed of multiple ANFIS models. Human perception based decision making ability and learning skill of ANFIS approach has been employed here to determine the required change in heading angle of underwater robot in horizontal as well as vertical plane for avoiding obstacles. A satisfactory navigational performance has been achieved for ANFIS based navigational strategy in simulation and experimental results.

- In chapter five, a new adaptive version of memetic evolution based optimization method has been proposed and implemented to achieve the near optimal safe path during underwater navigation. Fitness function has been designed by integrating

essential criteria for successful navigation like obstacle avoidance and path length minimization.

- In chapter six, a dynamic version of Differential Evolution has been formulated and employed as three-dimensional path planning algorithm. Proposed Dynamic Differential Evolution based optimization has stochastically found out the next global best position for underwater robot in each iteration while maintaining safe clearance from obstacles.

- In chapter seven, adaptive tuning of control parameters based on fitness of population and automatic selection of vector perturbation schemes for solution vectors have been introduced to improve local search ability of HS algorithm. Proposed adaptive version of HS has been employed as navigational strategy of underwater robot.

- Two simple metaheuristics, DE and HS, which are complementary for each other, have been integrated in chapter eight in a cooperative manner to enhance the convergence speed and search quality of three-dimensional path optimization process. Simulation and experimental investigations on proposed hybrid metaheuristics based navigational methodology have been executed in a realistic manner

- In chapter nine, evolutionary based stochastic computation and classical estimation techniques have been coupled together to incorporate some certainty and perfection within human perception based decision making abilities of ANFIS model in an automatic manner. Minimization of prediction error has been considered as major objective of the proposed hybrid learning approach for ANFIS model. Direction of underwater robot's motion has been accurately determined by employing the proposed ANFIS architecture associated with self-learning ability.

- In chapter ten, all proposed navigational strategies have been implemented for same simulation and experimental scenarios and performances have evaluated in terms of average error regarding path length and travel time. Over all, the errors between simulated and experimental performances have been recorded within the range of 4-7% in the present dissertation. In average, Hybrid of DE-HS approach and ANFIS trained with hybrid learning based on SFLA-RLS scheme have shown most desirable

performance by reducing error within the range of 4.5-5.5% for both path length and travel time.

11.3 Applications of Findings

The specific findings of present dissertation have already been described in previous section. The possible applications of all findings are mentioned below:

- Executed kinematic and dynamic modeling of the small size underwater robot will be of great help for the research communities to use such type of underwater robot for various marine applications like long term observations within sea area, stabilization of underwater robot in submerged condition, motion along different axes of underwater robot etc.

- Underwater robot embedded with any of the proposed navigational methodologies such as Manifold ANFIS approach, Adaptive Shuffled Frog Leaping Algorithm, Dynamic Differential Evolution, Adaptively Tuned Harmony Search and their hybridizations can be applied for surveillance of underwater scenarios, taking videos of rare views of sea floor, exploration of water mines in hazardous environment, monitoring of aqua lives etc.

11.4 Future Direction of Research

Extensive study has been conducted here to find out suitable navigational strategy for underwater robot. Still some research aspects of three-dimensional path planning algorithms can be considered in near future to achieve success various practical applications of underwater robot. Some of them have been mentioned here:

- Present research work has mainly focused on avoiding collision with static obstacles. Dynamic obstacles within navigational scenarios can be considered as future scope of recent investigation.

- In current study, start and goal positions of underwater robot have been considered to be static and known to navigational controller. The difficulties of research may be enhanced by introducing variable target positions within given scenarios.

- Navigation of multiple underwater robots can also be considered within future scope of research to meet the requirements of real-time naval applications.

References

1. Fossen, T. I., Marine Control Systems: Guidance. Navigation and Control of Ships, Rigs and Underwater Vehicles, *Marine Cybernetics*, Trondheim, Norway (2002).
2. Yuh, J., Design and control of autonomous underwater robots: A survey, *Autonomous Robots*, 8(1), (2000) 7-24.
3. Gonzalez, L. A., Design, modelling and control of an autonomous underwater vehicle, BE Thesis, *The University of Western Australia*, Australia (2004).
4. Isern-González, J., Hernández-Sosa, D., Fernández-Perdomo, E., Cabrera-Gámez, J., Domínguez-Brito, A. C., and Prieto-Marañón, V., Obstacle avoidance in underwater glider path planning, *Journal of Physical Agents*, 6(1), (2012) 11-20.
5. Ataei, M., and Yousefi-Koma, A., Three-dimensional optimal path planning for waypoint guidance of an autonomous underwater vehicle, *Robotics and Autonomous Systems*, 67, (2015) 23-32.
6. Geranmehr, B., and Nekoo, S. R., Nonlinear suboptimal control of fully coupled non-affine six-DOF autonomous underwater vehicle using the state-dependent Riccati equation, *Ocean Engineering*, 96, (2015) 248-257.
7. Antonelli, G., Fossen, T. I., and Yoerger, D. R., Underwater robotics, *Springer handbook of robotics*, Springer Berlin Heidelberg, (2008) 987-1008.
8. Brutzman, D. P., Dissertation: A Virtual World for an Autonomous Underwater Vehicle, *Naval Post Graduate School*, December, 2, (1994).
9. Jordán, M. A., and Bustamante, J. L., Adaptive control for guidance of underwater vehicles, *Vienna, Austria: In-Tech*, (2009) 251-278.
10. Yuh, J., Modeling and control of underwater robotic vehicles, *IEEE Transaction on Systems, Man, and Cybernetics*, 20(6), (1990) 1475-1483.
11. Li, Ye., Liu, J., and Shen, M., Dynamics model of underwater robot motion control in 6 degrees of freedom, *Journey of Harbin Institute of Technology*, 12(4), (2005) 456-459.
12. Antonelli, G., Caccavale, F., and Chiaverini, S., Adaptive tracking control of underwater vehicle-manipulator systems based on the virtual decomposition approach, *IEEE Transaction on Robotics and Automation*, 20(3), (2004) 594-602.

13. Nahon, M., A simplified dynamics model for autonomous underwater vehicles, *Symposium on Autonomous Underwater Vehicle Technology,* 2-6 June, 1996, Monterey, CA, 373-379.
14. Fischer, N., Hughes, D., Walters, P., Schwartz, E. M., and Dixon, W. E., Nonlinear RISE-based control of an autonomous underwater vehicle, *IEEE Transactions on Robotics*, 30(4), (2014) 845-852.
15. Dong, Z., Wan, L., Li, Y., Liu, T., Zhuang, J., and Zhang, G., Point Stabilization for an under actuated AUV in the Presence of Ocean Currents, *International Journal of Advanced Robotic Systems*, 12, (2015), DOI: 10.5772/61037.
16. Tsourveloudis, N. C., Doitsidis, L., and Valavanis, K. P., Autonomous navigation of unmanned vehicles: A fuzzy logic perspective, *Cutting Edge Robotics*, (2005) 291-310.
17. Bennett, A.A., and Leonard, J.J., A behavior-based approach to adaptive feature detection and following with autonomous underwater vehicles, *IEEE Journal of Oceanic Engineering*. 25(2), (2000) 213–226.
18. Breivik, M., and Fossen, T., Principles of guidance-based path following in 2D and 3D, *44th IEEE Conference on Decision and Control and European Control Conference. CDC-ECC'05*, 12-15 December, 2005, Seville, Spain, 627-634.
19. Seto, M. L., Paull, L., and Saeedi, S., Introduction to autonomy for marine robots. *Marine Robot Autonomy,* Springer New York, (2013) 1-46.
20. Fossen T I, Marine control systems: guidance, navigation and control of ships, rigs and underwater vehicles, *Marine Cybernetics AS*, Trondheim, (2002).
21. Santhakumar, M., and Asokan, T., Investigations on the hybrid tracking control of an under actuated autonomous underwater robot, *Advanced Robotics*, 24(11), (2010) 1529-1556.
22. LaValle, S.M., Planning algorithms, *Cambridge University Press*, New York (2006).
23. Khatib, O., Real-time obstacle avoidance for manipulators and mobile robots, *The International Journal of Robotics Research,* 5(1), (1986) 90-98.
24. Saravanakumar, S., and Asokan, T., Multipoint potential field method for path planning of autonomous underwater vehicles in 3D space, *Intelligent Service Robotics*, *6*(4), (2013) 211-224.

25. Miao, H., and Huang, X., A Heuristic Field Navigation Approach for Autonomous Underwater Vehicles*, Intelligent Automation and Soft Computing*, 20(1), (2014) 15-32.
26. Carlési, N., Michel, F., Jouvencel, B., and Ferber, J., Generic architecture for multi-AUV cooperation based on a multi-agent reactive organizational approach, *IEEE International Conference on Intelligent Robots and Systems (IROS),* 25-30 September, 2011, San Francisco, CA, 5041-5047.
27. McGann, C., Py, F., Rajan, K., Thomas, H., Henthorn, R., and McEwen, R., A deliberative architecture for AUV control, *IEEE International Conference on Robotics and Automation (ICRA 2008),* 19-23 May, 2008, Pasadena, CA, 1049-1054.
28. Teck, T. Y., Chitre, M., and Vadakkepat, P., Hierarchical agent-based command and control system for autonomous underwater vehicles, *International Conference on Autonomous and Intelligent Systems (AIS)*, 21-23 June, 2010, Povoa de Varzim, Portugal, 1-6.
29. Brooks, R., A robust layered control system for a mobile robot, *IEEE Journal of Robotics and Automation*, 2(1), (1986) 14-23.
30. Byrnes Jr, R. B., The Rational Behaviour Model: A Multi-Paradigm, Tri-Level Software Architecture for the Control of Autonomous Vehicles, Ph.D. Dissertation, *Naval Postgraduate School, Monterey Californi*a, March 1993.
31. Healey, A. J., and Lienard, D., Multivariable sliding mode control for autonomous diving and steering of unmanned underwater vehicles, *IEEE Journal of Oceanic Engineering*, 18(3), (1993) 327-339.
32. Budiyono, A., Robust Control Synthesis for an Unmanned Underwater Vehicle, *World Academy of Science, Engineering and Technology*, 60, (2009) 48-55.
33. Li, J. H., and Lee, P. M., Design of an adaptive nonlinear controller for depth control of an autonomous underwater vehicle, *Ocean Engineering*, 32(17), (2005) 2165-2181.
34. Lapierre, L., Robust diving control of an AUV, *Ocean Engineering*, 36(1), (2009) 92-104.
35. Krstic, M., Kokotovic, P. V., and Kanellakopoulos, I., Nonlinear and adaptive control design, *John Wiley and Sons, Inc.*, New York, USA (1995).
36. Subudhi, B., and Atta, D., Design of a Path Following Controller for an Underactuated AUV, *Archives of Control Sciences*, 19(3), (2009) 245-259.

37. Do, K. K., Practical Control of Underactuated Ships, *Ocean Engineering*, 37(13), (2010) 1111-1119.
38. Gao, B., Xu, D. M., and Yan, W. S., Framed-quadtree path planning for an underwater vehicle with the task of tracking a moving target, *Journal of Marine Science and Application*, 9(1), (2010) 27-33.
39. Santhakumar, M., and Kim, J., Robust adaptive tracking control of autonomous underwater vehicle-manipulator systems, *Journal of Dynamic Systems, Measurement, and Control*, 136(5), (2014) 054502 (10 pages).
40. Bian, X., Qu, Y., Yan, Z., and Zhang, W., Nonlinear feedback control for trajectory tracking of an unmanned underwater vehicle, *IEEE International Conference on Information and Automation (ICIA),* 20-23 June, 2010, Harbin, China, 1387-1392.
41. Repoulias, F., and Papadopoulos, E., Three dimensional trajectory control of under actuated AUVs, *European Control Conference (ECC),* 2-5 July, 2007, Kos, Greece, 3492-3499.
42. Wang, F., Wan, L., Su, Y. M., and Xu, Y. R., AUV modelling and motion control strategy design, *Journal of Marine Science and Application*, 9(4), (2010) 379-385.
43. Kim, H., and Cho, H., Numerical study on control derivatives of a high-speed underwater vehicle, *Journal of Mechanical Science and Technology*, 25(3), (2011) 759-765.
44. He-Ming, J., Wen-long, S., and Zi-yin, C., Nonlinear Backstepping Control of Underactuated AUV in Diving Plane, *Advances in Information Sciences and Service Sciences*, 4(9), (2012).
45. Yan, Z., Liu, Y., Zhou, J., and Wu, D., Path Following Control of an AUV under the Current Using the SVR-ADRC, *Journal of Applied Mathematics*, (2014) 476419 (12 pages).
46. Pousinho, H. M. I., Mendes, V. M. F., and Catalão, J. P. S., A hybrid PSO-ANFIS approach for short-term wind power prediction in Portugal, *Energy Conversion and Management*, 52(1), (2011) 397-402.
47. Khosla, A., Kumar, S., Aggarwal, K. K., and Singh, J., Particle swarm for fuzzy models identification, In *Swarm Intelligent Systems*, Springer Berlin Heidelberg, (2006) 149-173.

48. Kanakakis, V., Valavanis, K. P., and Tsourveloudis, N. C., Fuzzy-logic based navigation of underwater vehicles, *Journal of Intelligent and Robotic Systems*, 40(1), (2004) 45-88.
49. DiBitetto, P.A., Fuzzy logic for depth control of unmanned undersea vehicles, *IEEE Journal of Oceanic Engineering*, 20, (1995) 242-248.
50. Feijun, S., and Smith, S., Automatic design and optimization of fuzzy logic controllers for an autonomous underwater vehicle, *OCEANS MTS/IEEE*, 11-14 September, 2000, Providence, Rhode Island, 2, 829-834.
51. Kwiesielewicz, M., Piotrowski, W., and Sutton, R. Predictive versus fuzzy control of autonomous underwater vehicle, *IEEE International Conference on Methods and Models in Automation and Robotics*, 2001.
52. Akkizidis, I. S., and Roberts, G. N., Fuzzy Modelling and Fuzzy-Neuro Motion Control of an Autonomous Underwater Robot, *IEEE Conference on Advanced Motion Control,* 29 June-1 July, 1998, Coimbra, Portugal, 641-646.
53. Hassanein, O., Anavatti, Sreenatha G., and Ray, Tapabrata., Fuzzy Modeling and Control for Autonomous Underwater Vehicle, *5th International Conference on Automation, Robotics and Applications*, December 6-8, 2011, Wellington, New Zealand, 169-174.
54. Tahboub, Khaldoun K., and Al-Din, Munaf S. N., A Neuro-Fuzzy Reasoning System for Mobile Robot Navigation, *Jordan Journal of Mechanical and Industrial Engineering*, 3(1), (2009) 77-88.
55. Amjad, M., Ishaque, Kashif, Abdullah, S.S and Salam, Z., Single Input Fuzzy Logic Controller for Unmanned Underwater Vehicle, *Journal of Intelligent and Robotic Systems*, 59, (2010) 87–100.
56. Azar, A. T., Adaptive neuro-fuzzy systems, *Fuzzy systems*, (2010) 85-110.
57. Raimondi, F. M., and Melluso, M., Hierarchical fuzzy/lyapunov control for horizontal plane trajectory tracking of underactuated AUV, *IEEE International Symposium on Industrial Electronics (ISIE),* 4-7 July, 2010, Bari, Italy, 1875-1882.
58. Zhao, N., Xu, D., Gao, J., and Yan, W., Fuzzy Behavioral Navigation for Bottom Collision Avoidance of Autonomous Underwater Vehicles, *Intelligent Robotics and Applications*, Springer Berlin Heidelberg, (2008) 122-130.

59. Liu, S., Wei, Y., and Gao, Y., 3D path planning for AUV using fuzzy logic, *International Conference on Computer Science and Information Processing (CSIP)*, 24-26 August, 2012, Xian, Shaanxi, 599-603.
60. Tang, Z., Liu, P., Luo, Q., Luo, J., and Xie, S., A Novel Control Method for Shallow Underwater Robot, *International Journal of Multimedia and Ubiquitous Engineering*, 10(3), (2015) 245-256.
61. Haykin, S., and Network, N., A comprehensive foundation, Neural Networks, 2, (2004).
62. Van de Ven, P. W. J., Flanagan, C., and Toal, D., Neural network control of underwater vehicles, *Engineering Applications of Artificial Intelligence*, 18(5), 533-547(2005).
63. Forouzantabar, A., Gholami, B., and Azadi, M., Adaptive Neural Network Control of Autonomous Underwater Vehicles, *International Scholarly and Scientific Research and Innovation*, World Academy of Science, Engineering and Technology, 6(7), (2012) 304-309.
64. Venugopal, K. P., Sudhakar, R., and Pandya, A. S., On-line learning control of autonomous underwater vehicles using feedforward neural networks, IEEE Journal of *Oceanic Engineering*, 17(4), (1992) 308-319.
65. Liang, X., Gan, Y., and Wan, L., Motion Controller for Autonomous Underwater Vehicle Based on Parallel Neural Network, *International Journal of Digital Content Technology and its Applications*, 4 (9), (2010) 61-67.
66. Li, J. H., Lee, P. M., and Lee, S. J., Motion control of an AUV using a neural network adaptive controller, *International Symposium on Underwater Technology*, 16-19 April, 2002, Tokyo, Japan, 217-221.
67. Lewis, F., Jagannathan, S. and Yesilderek, A., Neural Network Control of Robot Manipulators and Nonlinear Systems, *Taylor and Francis*, Inc., Bristol, PA, USA, 1998.
68. Jagannathan, S., and Galan, G., One-Layer Neural-Network Controller With Pre-processed Inputs for Autonomous Underwater Vehicles, *IEEE Transactions on Vehicular Technology*, 52(5), (September, 2003) 1342-1355.
69. Amin, R., Khayyat, A. A., and Osgouie, K. G., Neural networks modeling of autonomous underwater vehicle, *IEEE/ASME International Conference on*

Mechatronics and Embedded Systems and Applications (MESA), 15-17 July 2010, Qingdao, Shandong, 14-19.

70. Zhou, J., Tang, Z., Zhang, H., and Jiao, J., Spatial Path Following for AUVs Using Adaptive Neural Network Controllers, *Mathematical Problems in Engineering*, (2013) 749689 (9 pages).
71. Hagras, H., and Sobh. T., Intelligent Learning and Control of Autonomous Robotic Agents Operating in Unstructured Environments, *Information Sciences*, 145(1-2), (August 2002) 1-12.
72. Babu, S. S., Kumar, C. S., and Faruqi, M. A., A Neural Network Online Controller for Autonomous Underwater Vehicle, *IEEE International Conference on Industrial Technology (ICIT)*, 15-17 December 2006, Mumbai, 2320-2324.
73. Ge, H., Jing, Z. L., and Gao, J. Neural network H-infinity robust adaptive control for autonomous underwater vehicle in 3-dimensional path following, *Control Theory and Applications*, 29(3), (2012) 317-322.
74. Bian, X., Zhou, J., Yan, Z., and Jia, H., Adaptive neural network control system of path following for AUVs, *Proceedings of IEEE Southeastcon,* 15-18 March, 2012, Orlando, Florida, USA, 1-5.
75. Sun, B., Zhu, D., Ding, F., and Yang, S. X., A novel tracking control approach for unmanned underwater vehicles based on bio-inspired neurodynamics, *Journal of marine science and technology*, 18(1), (2013) 63-74.
76. Park, B. S., Neural Network-Based Tracking Control of Under actuated Autonomous Underwater Vehicles with Model Uncertainties, *Journal of Dynamic Systems, Measurement, and Control*, 137(2), (2014) 021004 (7pages).
77. Jang, J. S. R., ANFIS: adaptive-network-based fuzzy inference system, *IEEE Transactions on Systems, Man and Cybernetics*, 23(3), (1993) 665-685.
78. Lee, C. S. G. Wang, J-S. and Yuh, J., Self-adaptive neuro-fuzzy systems for autonomous underwater vehicle control, *Advanced Robotics*, 15(5), (2001) 589-608.
79. Wang, J-S. and Lee, C. S. G., Self-Adaptive Recurrent Neuro-Fuzzy Control of an Autonomous Underwater Vehicle, *IEEE Transactions on Robotics and Automation*, 19(2), (April 2003) 283-295.
80. Kim, T. W. and Yuh, J., Application of online neuro-fuzzy controller to AUVs, *Information Sciences*, 145, (2002) 169-182.

81. Woo, P. Y., and Polisetty, V., ANFIS Generated Dynamic Path Planning for a Mobile Robot to Track a Randomly Moving Target in a 3-D Space with Obstacle Avoidance, *IEEE International Conference on Fuzzy Systems* (*FUZZ*), 18-23 July, 2010, Barcelona, Spain, 1-8.
82. Cherroun, L., and Boumehraz, M., Path following behavior for an autonomous mobile robot using neuro-fuzzy controller, *International Journal of System Assurance Engineering and Management*, 5(3), (September 2014) 352-360.
83. Salman, S. A., Anavatti, Sreenatha A., and Asokan, T., Adaptive fuzzy control of unmanned underwater vehicles, *Indian Journal of Geo-Marine Sciences*, 40(2), (April 2011) 168-175.
84. Chen, H., Zheng, W., and Gai, X., Neuro-fuzzy control of underwater robot based on disturbance compensation, *8th World Congress on Intelligent Control and Automation (WCICA)*, 7-9 July, 2010, Jinan, China, 413-418.
85. Shi, X., Chen, J., Yan, Z., and Li, T., Design of AUV Height Control Based on Adaptive Neuro-Fuzzy Inference System, *IEEE International Conference on Information and Automation*, 20-23 June , 2010, Harbin, China,1646-1651.
86. Hassanein, O., Anavatti, S. G., and Ray, T., On-Line Adaptive Fuzzy Modeling and Control for Autonomous Underwater Vehicle, *Recent Advances in Robotics and Automation,* Series in Studies in Computational Intelligence, 480, (2013) 57-70.
87. Ch, S., and Mathur, S., Modelling uncertainty analysis in flow and solute transport model using Adaptive Neuro Fuzzy Inference System and particle swarm optimization, *KSCE Journal of Civil Engineering*, 14(6), (2010) 941-951.
88. Grosan, C., and Abraham, A., Hybrid evolutionary algorithms: methodologies, architectures, and reviews, *Hybrid evolutionary algorithms,* Studies in Computational Intelligence Springer Berlin Heidelberg, 75, (2007) 1-17.
89. Alvarez, A., Caiti, A., and Onken, R., Evolutionary path planning for autonomous underwater vehicles in a variable ocean, *IEEE Journal of Oceanic Engineering*, 29(2), (2004) 418-429.
90. Sugihara, K., and Yuh, J., GA-based motion planning for underwater robotic vehicles, 10th *International Symposium on Unmanned Untethered Submersible Technology,* September, 1997, University of New Hampshire-Marine Systems, 406-415.

91. Cheng, C. T., Fallahi, K., Leung, H., and Tse, C. K., A genetic algorithm-inspired UUV path planner based on dynamic programming, *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews,* 42(6), (2012) 1128-1134.
92. Tanakitkorn, K., Wilson, P., Turnock, S. R., and Phillips, A. B., Grid-based GA path planning with improved cost function for an over-actuated hover-capable AUV, *IEEE/OES on Autonomous Underwater Vehicles*, 6-9 October, 2014, Oxford, Mississippi, 1-8.
93. Hong-jian, W., Xin-Qian, B., Jie, Z., Fu-Guang, D., and Guo-Qing, X., A GA path planner based on domain knowledge for AUV, *OCEAN'04 MTTS/IEEE TECHNO*, 9-12 November, 2004, Kobe, Japan, 3, 1570-1573.
94. Sun, Y., and Zhang, R., Research on global path planning for AUV based on GA, *Mechanical Engineering and Technology,* Advances in Intelligent and Soft Computing, Springer Berlin Heidelberg, 125, (2012) 311-318.
95. Yan, G., Wang, L., Zhou, J., and Zha, Z., Path Planning Based on Improved Genetic Algorithm for AUV, *Journal of Chongqing University of Technology (Natural Science)*, 5, (2010) 115-120.
96. Sun, J., and Wu, S., Route planning of cruise missile based on improved particle swarm algorithm, *Journal of Beijing University of Aeronautics and Astronautics*, 37(10), (2011) 1228-1232.
97. Fan, Z. C., Path planning method based on the algorithm of PSO and elastic rope for underwater vehicle in three-dimensional space [Master thesis], Harbin Engineering University, 2013.
98. Tang, X., Yu, F., and Chen, R., Path planning of underwater vehicle based on particle swarm optimization, *International Conference on Intelligent Control and Information Processing (ICICIP)*, 13-15 August, 2010, Dalian, China, 123-126.
99. Yan, Z., Deng, C., Chi, D., Chen, T., and Hou, S., Path Planning Method for UUV Homing and Docking in Movement Disorders Environment, *The Scientific World Journal*, (2014) 246469 (13 pages).
100. Yan, Z., Deng, C., Li, B., and Zhou, J., Novel particle swarm optimization and its application in calibrating the underwater transponder coordinates, *Mathematical Problems in Engineering*, (2014) 672412 (12pages).

101. Zeng, Z., Sammut, K., Lammas, A., He, F., and Tang, Y., Efficient path re-planning for AUVs operating in spatiotemporal currents, *Journal of Intelligent and Robotic Systems*, 79(1), (2014) 135-153.
102. Amiri, Z., Pouyan, A. A., and Mashayekhi, H., A topology control algorithm for autonomous underwater robots in three-dimensional space using PSO, *Journal of AI and Data Mining*, 3(2), (2015) 191-201.
103. Wang, P., Meng, P., and Ning, T., Path Planning Based on Hybrid Adaptive Ant Colony Algorithm for AUV, *11th International Symposium on Distributed Computing and Applications to Business, Engineering & Science (DCABES)*, 19-22 October 2012, Guilin, China, 157-160.
104. Liu, L. Q., Dai, Y. T., Wang, L. H., and Gan, X. L., Research on Global Path Planning of Underwater Vehicle Based on Ant Colony Algorithm, *Journal of System Simulation*, 19(18), (2007) 4174-4177.
105. Liu, L. Q., Yu, F., and Dai, Y. T., Path planning of underwater vehicle in 3D space based on ant colony algorithm, *Journal of System Simulation*, 20(14), (2008) 3712-3716.
106. Wang, H. J., and Xiong, W., Research on global path planning based on ant colony optimization for AUV, *Journal of Marine Science and Application*, 8(1), (2009) 58-64.
107. Liu, L., and Dai, Y., 3D space path planning of complex environmental underwater vehicle, *International Joint Conference on Computational Sciences and Optimization*, 24-26 April 2009, Sanya, Hainan, 2, 204-209.
108. Eusuff, M. M., and Lansey, K. E., Optimization of water distribution network design using the shuffled frog leaping algorithm, *Journal of Water Resources Planning and Management*, 129(3), (2003) 210-225.
109. Eusuff, M., Lansey, K., and Pasha, F., Shuffled frog-leaping algorithm: a memetic meta-heuristic for discrete optimization. *Engineering Optimization*, 38(2), (2006) 129-154.
110. Li, J., Pan, Q., and Xie, S., An effective shuffled frog-leaping algorithm for multi-objective flexible job shop scheduling problems, *Applied Mathematics and Computation*, 218(18), (2012) 9353-9371.

111. Venkatesan, T., and Sanavullah, M. Y., SFLA approach to solve PBUC problem with emission limitation, *International Journal of Electrical Power and Energy Systems*, 46, (2013) 1-9.
112. Elbeltagi, E., Hegazy, T., and Grierson, D., A modified shuffled frog-leaping optimization algorithm: applications to project management, *Structure and Infrastructure Engineering*, 3(1), (2007) 53-60.
113. Amiri, B., Fathian, M., and Maroosi, A., Application of shuffled frog-leaping algorithm on clustering, *Journal of Advanced Manufacturing Technology*, 45(1-2), (2009) 199-209.
114. Vahed, R. A., and Mirzaei, A.H., A hybrid multi-objective shuffled frog-leaping algorithm for a mixed-model assembly line sequencing problem, Computer and Industrial Engineering, 53(4), (2007) 642–66.
115. Hassanzadeh, I., Madani, K., and Badamchizadeh, M. A., Mobile robot path planning based on shuffled frog leaping optimization algorithm, *IEEE Conference on Automation Science and Engineering (CASE),* 21-24 August, 2010, Toronto, Canada, 680-685.
116. Afzalan, E., Taghikhani, M. A., and Sedighizadeh, M., Optimal placement and sizing of DG in radial distribution networks using SFLA, *International Journal of Energy Engineering*, 2(3), (2012) 73-77.
117. Hidalgo-Paniagua, A., Vega-Rodríguez, M. A., Ferruz, J., and Pavón, N., MOSFLA-MRPP: Multi-Objective Shuffled Frog-Leaping Algorithm applied to Mobile Robot Path Planning, *Engineering Applications of Artificial Intelligence*, 44, (2015) 123-136.
118. Sarker, R., and Abbass, H. A., Differential Evolution for Solving Multi-objective Optimization Problems, *Asia-Pacific Journal of Operational Research,* 21(2), (2004) 225-240.
119. Civicioglu, P., and Besdok, E., A conceptual comparison of the Cuckoo-search, particle swarm optimization, differential evolution and artificial bee colony algorithms, *Artificial Intelligence Review*, 39(4), (April 2013) 315-346.
120. Price, K., Storn, R., and Lampinen, J., Differential Evolution: A Practical Approach to Global Optimization, *In Natural Computing Series*, Springer Verlag Berlin, Germany, 2005.

121. Coelho, L. d S., Nedjah N., and Mourelle, L. de M., Differential Evolution Approach Using Chaotic Sequences Applied to Planning of Mobile Robot in a Static Environment with Obstacles, Mobile Robots: The evolutionary Approach, *Studies in Computational Intelligence,* 50 (2007) 3-22.
122. Wang, X. and Xu, G., Robot Path Planning Based on Chaos Concise Differential Evolution and RFNN Control, *The Open Automation and Control Systems Journal*, 6 (2014) 69-76.
123. Chakraborty, J., Konar A., Chakraborty, U. K., and Jain, L.C., Distributed Cooperative Multi-Robot Path Planning Using Differential Evolution, *Evolutionary IEEE World Congress on Computational Intelligence*, 1-6 June 2008, Hong Kong, China, 718-725.
124. Mo, H., and Meng, L., Robot Path Planning Based on Differential Evolution in Static Environment, *International Journal of Digital Content Technology and its Applications,* 6(20), (November 2012) 122-129.
125. Bashiri, M., Vatankhah, H., and Ghidary, S. S., Hybrid adaptive differential evolution for mobile robot localization, *Intelligent Service Robotics*, 5, (2012) 99-107.
126. Nikolos, I. K., and Tsourveloudis, N. C., Path Planning for Cooperating Unmanned Vehicles over 3-D Terrain, *Informatics in Control Automation and Robotics*, Lecture Notes in Electrical Engineering, 24, (2009) 153-168.
127. Zhang, X., and Duan, H., An improved constrained differential evolution algorithm for unmanned aerial vehicle global route planning, *Applied Soft Computing,* 26, (2015) 270-284.
128. Zamuda, A., and Sosa, J.D.H., Differential evolution and underwater glider path planning applied to the short-term opportunistic sampling of dynamic mesoscale ocean structures, *Applied Soft Computing,* 24, (2014) 95–108.
129. Alia, O. M., and Mandava, R., The variants of the harmony search algorithm: an overview, *Artificial Intelligence Review*, 36 (1), (2011) 49-68.
130. Mahdavi, M., Fesanghary, M., and Damangir, E., An improved harmony search algorithm for solving optimization problems, *Applied Mathematics and Computation*, 188(2), (2007) 1567-1579.
131. Omran, M. G. H., and Mahdavi, M., Global-best harmony search, *Applied Mathematics and Computation*, 198(2), (2008) 643-656.

132. Pan, Q.K., Suganthan, P. N., Tasgetiren, M. F., and Liang, J. J., A self-adaptive global best harmony search algorithm for continuous optimization problems, *Applied Mathematics and Computation*, 216(3), (2010) 830-848.
133. Pan, Q.K., Suganthan, P. N., Liang, J. J., and Tasgetiren, M. F., A local-best harmony search algorithm with dynamic subpopulations, *Engineering Optimization*, 42(2), (2010)101-117.
134. Wang, C. M., and Huang, Y. F., Self-adaptive harmony search algorithm for optimization, *Expert Systems with Applications*, 37(4), (2010) 2826-2837.
135. Zou, D., Gao, L., Wu, J., and Li, S., Novel global harmony search algorithm for unconstrained problems, *Neuro-computing*, 73(16-18), (2010) 3308-3318.
136. Das, S., Mukhopadhyay, A., Roy, A., Abraham, A., and Panigrahi, B. K., Exploratory power of the harmony search algorithm: analysis and improvements for global numerical optimization, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 41(1), (2011) 89-106.
137. Yadav, P., Kumar, R., Panda, S. K., and Chang, C. S., An intelligent tuned harmony search algorithm for optimisation, *Information Sciences*, 196, (2012) 47-72.
138. Chen, J., Pan, Q.K, and Li, J.Q, Harmony search algorithm with dynamic control parameters, *Applied Mathematics and Computation*, 219(2), (2012) 592-604.
139. El-Abd, M., An improved global-best harmony search algorithm, *Applied Mathematics and Computation*, 222, (2013) 94-106.
140. Xiang, W. L., An, M. Q., Li, Y. Z., He, R. C., and Zhang, J. F., An improved global-best harmony search algorithm for faster optimization, *Expert Systems with Applications*, 41(13), (2014) 5788-5803.
141. Geem, Z. W., Lee, K. S., and Park, Y., Application of harmony search to vehicle routing, *American Journal of Applied Sciences*, 2(12), (2005) 1552-1557.
142. Jati, A., Singh, G., Rakshit, P., Konar, A., Kim, E., and Nagar, A. K., A hybridisation of Improved Harmony Search and Bacterial Foraging for multi-robot motion planning, *IEEE Congress on Evolutionary Computation (CEC)*, 10-15 June, 2012, Brisbane, Australia, 1-8.
143. Panov, S., and Koceska, N., Global Path Planning in Grid-Based Environments Using Novel Metaheuristic Algorithm, *ICT Innovations, Advances in Intelligent Systems and Computing*, Springer International Publishing, 231, (2013) 121-130.

144. Sharma, K. D., Chatterjee, A., and Rakshit, A., Harmony search-based hybrid stable adaptive fuzzy tracking controllers for vision-based mobile robot navigation, *Machine Vision and Applications*, 25(2), (2014) 405-419.

145. Mirkhani, M., Forsati, R., Shahri, A. M., and Moayedikia, A., A novel efficient algorithm for mobile robot localization, *Robotics and Autonomous Systems*, 61(9), (2013) 920-931.

146. Li, G., and Liu, Z, Underwater Penetration Path Planning Based on CHS Algorithm, *Seventh International Joint Conference on Computational Sciences and Optimization (CSO),* 4-6 July 2014, Beijing, China, 199-203.

147. Sinha, A., and Goldberg, D. E., A survey of hybrid genetic and evolutionary algorithms, Illinois Genetic Algorithms Laboratory *report No.*, *2003004* (2003).

148. Zhang, L., Pang, Y., Su, Y., and Liang, Y., HPSO-based fuzzy neural network control for AUV, *Journal of Control Theory and Applications*, 6(3), (2008) 322-326.

149. Fossen, T. I., Guidance and control of ocean vehicles, *John Wiley and Sons, Inc*., (1994).

150. Repoulias, F., and Papadopoulos, E., Planar trajectory planning and tracking control design for underactuated AUVs, *Ocean Engineering*, 34(11), (2007) 1650-1667.

151. Yager, R. R., and Zadeh, L. A., Fuzzy sets, neural networks and soft computing, *John Wiley and Sons, Inc.,* (1994).

152. Yager, R. R., and Zadeh, L. A., An introduction to fuzzy logic applications in intelligent systems, *The Springer International Series in Engineering and Computer Science*, Springer Science and Business Media, 165, (2012).

153. Pradhan, S. K., Parhi, D. R., and Panda, A. K., Fuzzy logic techniques for navigation of several mobile robots, *Applied soft computing*, *9*(1), 290-304 (2009).

154. Parhi, D. R., and Singh, M. K., Navigational path analysis of mobile robots using an adaptive neuro-fuzzy inference system controller in a dynamic environment, *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 224(6), (2010) 1369-1381.

155. Demet, Ö., A Behavior based Robot Control System using Neuro-Fuzzy Approach, *Ph.D. dissertation*, The Graduate School of Natural and Applied Sciences of the Middle East Technical University, December (2003).

156. Yuan, H. and Qu, Z., Optimal real-time collision-free motion planning for autonomous underwater vehicles in a 3D underwater space, *IET Control Theory Application.*, 3(6), 2009 712-721.
157. Li, S. and Guo, Y., Neural-Network Based AUV Path Planning in Estuary Environments, *10th World Congress on Intelligent Control and Automation (WCICA*), 6-8 July, 2012, Beijing, China, 3724-3730.
158. Elbeltagi, E., Hegazy, T., and Grierson, D., Comparison among five evolutionary-based optimization algorithms, *Advanced Engineering Informatics*, 19(1), (2005) 43-53.
159. Li, X., Luo, J., Chen, M. R., and Wang, N., An improved shuffled frog-leaping algorithm with extreme optimization for continuous optimization, *Information Sciences*, 192, (2012) 143-151.
160. Jadidoleslam, M., and Ebrahimi, A, Reliability constrained generation expansion planning by a modified shuffled frog leaping algorithm*, International Journal of Electrical Power and Energy Systems*, 64, (2015) 743-751.
161. Aghdam, K. M., Mirzaee, I., Pourmahmood, N., and Aghababa, M. P., Adaptive Mutated Momentum Shuffled Frog Leaping Algorithm for Design of Water Distribution Networks, *Arabian Journal for Science and Engineering*, 39(11), (2014) 7717-7727.
162. Sarkheyli, A., Zain, A. M., and Sharif, S., The role of basic, modified and hybrid shuffled frog leaping algorithm on optimization problems: a review, *Soft Computing*, (2014) 1-28.
163. Bhattacharjee, K. K., and Sarmah, S. P., Shuffled frog leaping algorithm and its application to 0/1 knapsack problem, *Applied Soft Computing*, 19, (2014) 252-263.
164. Zhu, G. Y., and Zhang, W. B., An improved Shuffled Frog-leaping Algorithm to optimize component pick-and-place sequencing optimization problem, *Expert Systems with Applications*, 41(15), (2014) 6818-6829.
165. Che, H., Li, C., He, X., and Huang, T., An intelligent method of swarm neural networks for equalities-constrained nonconvex optimization, *Neurocomputing*, 167, (2015) 569-177.
166. Luo, J. P., Li, X., and Chen, M. R., Hybrid shuffled frog leaping algorithm for energy-efficient dynamic consolidation of virtual machines in cloud data centers, *Expert Systems with Applications*, 41(13), (2014) 5804-5816.

167. Perez, I., Gomez-Gonzalez, M., and Jurado, F., Estimation of induction motor parameters using shuffled frog-leaping algorithm, *Electrical Engineering*, 95(3), (2013) 267-275.
168. Aghababa, M. P., 3D path planning for underwater vehicles using five evolutionary optimization algorithms avoiding static and energetic obstacles, *Applied Ocean Research*, 38, (2012) 48-62.
169. Kwon, K. Y., Cho, J., Yoo, B. S., and Joh, J., Autonomous navigation of an underwater robot using fuzzy logic, *IEEE Annual Meeting of the North American Fuzzy Information Processing Society (NAFIPS)*, 26-28 June, 2005, Detroit, Michigan, 540-545.
170. Liu, C., Zhao, Y., Gao, F., and Liu, L., Three-Dimensional Path Planning Method for Autonomous Underwater Vehicle Based on Modified Firefly Algorithm, *Mathematical Problems in Engineering,* (2015) 561394 (10pages).
171. Ali, M. M., and Törn, A., Population set-based global optimization algorithms: some modifications and numerical studies. *Computers and Operations Research*, 31(10), (2004) 1703-1725.
172. Ali, M. M., Differential evolution with preferential crossover, *European Journal of Operational Research*, 181(3), (2007) 1137-1147.
173. Gamperle, R., Müller, S. D., and Koumoutsakos, P., A parameter study for differential evolution, WSEAS *International Conference on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation*, 11-15 February, 2002, Interlaken, Switzerland, 293-298.
174. Ronkkonen J., Kukkonen S. and Price K. V., Real parameter optimization with differential evolution, *IEEE Congress on Evolutionary Computation*, 2-5 September, 2005, Edinburgh, Scotland, 1, 506-513.
175. Kaelo, P. and Ali, M.M., A numerical study of some modified differential evolution algorithms, *European Journal of Operational Research*, 169, (2006) 1176-1184.
176. Karaboga, D., and Akay, B., A comparative study of artificial bee colony algorithm, *Applied Mathematics and Computation,* 214(12), (2009) 108-132.
177. Zaharie, D., Influence of crossover on the behaviour of differential evolution algorithms, *Applied Soft computing*, 9(3), (2009) 1126-1138.

178. Storn, R., and Price, K., Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, *Journal of global optimization*, 11(4), (1997) 341-359.
179. Brest, J., Greiner, S., Bošković, B., Mernik, M., and Zumer, V., Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, *IEEE Transactions on Evolutionary Computation,* 10(6), 646-657 (2006).
180. Kachitvichyanukul, V., Comparison of Three Evolutionary Algorithms, *Industrial Engineeering and Management Systems*, 11(3), (2012) 215-223.
181. Das, S., and Suganthan, P. N. Differential evolution: a survey of the state-of-the-art, *IEEE Transactions on Evolutionary Computation*, 15(1), 4-31 (2011).
182. Cheng, J., Zhang, G., and Neri, F., Enhancing distributed differential evolution with multicultural migration for global numerical optimization, *Information Sciences*, 247, (2013) 72-93.
183. Brest, J., Bošković, B., Greiner, S., Žumer, V., and Maučec, M. S., Performance comparison of self-adaptive and adaptive differential evolution algorithms, *Soft Computing*, 11(7), (2007) 617-629.
184. Zhang, J., and Sanderson, A. C., JADE: adaptive differential evolution with optional external archive, *IEEE Transactions on Evolutionary Computation*, 13(5), (2009) 945-958.
185. Qin, A. K., Huang, V. L., and Suganthan, P. N., Differential evolution algorithm with strategy adaptation for global numerical optimization, *IEEE Transactions on Evolutionary Computation*, 13(2), (2009) 398-417.
186. Pan, Q. K., Suganthan, P. N., Wang, L., Gao, L., and Mallipeddi, R., A differential evolution algorithm with self-adapting strategy and control parameters, *Computers and Operations Research*, 38(1), (2011) 394-408.
187. Jiang, L. and Zhu, D., Three-Dimensional Path Planning for AUV Based on Fuzzy Control, *Chinese Intelligent Automation Conference*, 23-25 August, 2013, Yangzhou, China, Lecture Notes in Electrical Engineering, 254, 31-40.
188. Niu, Q., Zhang, H., Li, K., and Irwin, G. W., An efficient harmony search with new pitch adjustment for dynamic economic dispatch, *Energy*, 65, (2014) 25-43.
189. Wang, G., and Guo, L., A novel hybrid bat algorithm with harmony search for global numerical optimization, *Journal of Applied Mathematics*, (2013) 696491 (21pages).

190. Pandi, V. R., and Panigrahi, B. K., Dynamic economic load dispatch using hybrid swarm intelligence based harmony search algorithm, *Expert Systems with Applications*, 38(7), (2011) 8509-8514.
191. Chakraborty, P., Roy, G. G., Das, S., Jain, D., and Abraham, A., An Improved Harmony Search Algorithm with Differential Mutation Operator, *Fundamenta Informaticae*, 95(4), (2009), 401-426.
192. Geem, Z. W., Kim, J. H., and Loganathan, G. V., A new heuristic optimization algorithm: harmony search, *Simulation*, 76(2), (2001) 60-68.
193. Guanglei, Z., and Heming, J., 3D path planning of AUV based on improved ant colony optimization, *32nd Chinese Control Conference (CCC)*, 26-28 July, 2013, Xian, China, 5017-5022.
194. Qin, A. K., and Forbes, F., Harmony search with differential mutation based pitch adjustment, *13th Annual conference on Genetic and evolutionary computation*, 12-16 July, 2011, Dublin, Ireland, 545-552.
195. Neri, F., and Tirronen, V., Recent advances in differential evolution: a survey and experimental analysis, *Artificial Intelligence Review*, 33(1-2), (2010) 61-106.
196. Liu, X., Cai, Z., and Yu, C., A hybrid harmony search approach based on differential evolution, *Journal of Information and Computational Science*, 8(10), (2011) 1889-1900.
197. Li, M., Wang, D. B., Bai, T. T., and Sheng, S. Z., Route planning based on particle swarm optimization with threat heuristic, *Electronics Optics and Control*, 18(12), (2011) 1-4.
198. Mallipeddi, R., Harmony search based parameter ensemble adaptation for differential evolution, *Journal of Applied Mathematics*, (2013) 750819 (12pages).
199. Shoorehdeli, M. A., Teshnehlab, M., and Sedigh, A. K., Training ANFIS as an identifier with intelligent hybrid stable learning algorithm based on particle swarm optimization and extended Kalman filter, *Fuzzy Sets and Systems*, 160(7), (2009) 922-948.
200. Wu, S., and Er, M. J., Dynamic fuzzy neural networks-a novel approach to function approximation, *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 30(2), (2000) 358-364.

201. Ghomsheh, V. S., Shoorehdeli, M. A., and Teshnehlab, M., Training ANFIS structure with modified PSO algorithm, *Mediterranean Conference on Control and Automation*, 27-29 June, Athens, Greece, 2007, 1-6.
202. Ghanem, W. A. H., and Jantan, A., Using Hybrid Artificial Bee Colony Algorithm and Particle Swarm Optimization for Training Feed-Forward Neural Networks, *Journal of Theoretical and Applied Information Technology*, 67(3), (2014) 664-674.
203. Sarangi, P. P., Sahu, A., and Panda, M., A Hybrid Differential Evolution and Back-Propagation Algorithm for Feed-forward Neural Network Training, *International Journal of Computer Applications*, 84(14), (2013) 1-9.
204. Khodashinskii, I. A., and Dudin, P. A., Identification of fuzzy systems using a continuous ant colony algorithm, *Optoelectronics, Instrumentation and Data Processing*, 48(1), (2012) 54-61.
205. Shoorehdeli, M. A., Teshnehlab, M., and Sedigh, A. K., Identification using ANFIS with intelligent hybrid stable learning algorithm approaches, *Neural Computing and Applications,* 18(2), (2009) 157-174.
206. Jiang, H. M., Kwong, C. K., Ip, W. H., and Wong, T. C., Modelling customer satisfaction for new product development using a PSO-based ANFIS approach, *Applied Soft Computing*, 12(2), (2012) 726-734.
207. Yan, M., Zhu, D., and Yang, S. X., A novel 3-d bio-inspired neural network model for the path planning of an auv in underwater environments, *Intelligent Automation and Soft Computing*, 19(4), (2013) 555-566.

List of Publications

International Journals:

1. Shubhasri Kundu and Dayal R. Parhi, "Reactive Navigation of Underwater Mobile Robot using ANFIS Approach in a Manifold Manner", *International Journal of Automation and Computing*, Springer (Accepted).
2. Dayal R. Parhi and Shubhasri Kundu, "Navigational control of underwater mobile robot using dynamic differential evolution approach", IMeche, *Part M: Journal of Engineering for the Maritime Environment*, SAGE (Accepted).
3. Shubhasri Kundu and Dayal R. Parhi, "Navigation of Underwater Robot based on Dynamically Adaptive Harmony Search Algorithm", *Memetic Computing*, Springer (Accepted).
4. Shubhasri Kundu and Dayal R. Parhi, "Differentially Adaptive Harmony Search for Path Planning of Underwater Robot", *Intelligent Service Robotics*, Springer (under review).
5. Dayal R. Parhi and Shubhasri Kundu, "Multiple ANFIS Model Approach for Motion Planning of Underwater Mobile Robot", *International Journal of Artificial Intelligence and Computational Research*, 6(1), 57-67 (2014).
6. Dayal R. Parhi and Shubhasri Kundu, "Analysis of Thruster Dynamics for Underwater Mobile Robot", *International Journal of Applied Artificial Intelligence in Engineering System*, 5(2), 129-139 (2013).
7. Shubhasri Kundu, Dayal R. Parhi and B.B.V.L Deepak, "Fuzzy-Neuro based Navigational Strategy for Mobile Robot", *International Journal of Scientific & Engineering Research*, 3(6), 97-102 (2012).

International Conferences:

1. Shubhasri Kundu and Dayal R. Parhi, "Modified Shuffled Frog Leaping Algorithm based 6DOF Motion for Underwater Mobile Robot", *International Conference on Computational Intelligence: Modeling, Techniques and Applications* (CIMTA- 2013), Procedia Technology 10 (2013), 26-28th September, 2013, Kalyani University, 295-303.
2. Shubhasri Kundu and Dayal R. Parhi, "Behavioural Analysis for Underwater Mobile Robot Navigation based on Shuffled Frog Leaping Algorithm", *Second International Conference on Intelligent Robotics, Automation and Manufacturing* (IRAM 2013), 16-19th December, 2013, IIT Indore, Emerald Group Publishing Limited, 44-50.
3. Shubhasri Kundu and Dayal R.Parhi, "Navigational Analysis for Underwater Robot Based on Multiple ANFIS Approach", *IEEE International Conference on Recent*

Advances in Mechanical Engineering and Interdisciplinary Developments (ICRAMID), 7-8th March, 2014, Nagercoil, Tamilnadu, 397-401.

4. Shubhasri Kundu, Chinmaya Sahu and Dayal R.Parhi, "ANFIS Approach based on Hybrid Learning for Motion Planning of Underwater Mobile Robot", *IEEE sponsored International Conference on Convergence of Technology* (I2CT 2014), 6-8th April, 2014, Pune, 1-6.
5. Shubhasri, K., and D. R. Parhi: "Navigation Based on Adaptive Shuffled Frog-Leaping Algorithm for Underwater Mobile Robot", *Intelligent Computing, Communication and Devices*, Springer India, ICCD-2014, 18-19th April, 2014, SOA University, Bhubaneswar, pp. 651-659.
6. Shubhasri Kundu and Dayal R.Parhi, "Underwater Mobile Robot Navigation based on Dynamically Modified Shuffled Frog Leaping Algorithm", *International Conference on Communication and Computing (ICC- 2014),* 12-14th June, 2014, Bangalore, 73-80.
7. Shubhasri Kundu, Manu Mishra and Dayal R.Parhi, "Autonomous navigation of underwater mobile robot based on harmony search optimization", *IEEE International Conference on Power Electronics, Drives and Energy Systems* (PEDES-2014), 16-19th December, 2014, IIT Mumbai, 1-6.

Patent:

Underwater Mobile Robot for Monitoring and Exploration Purpose (Application No.: 359/KOL/2015), Inventors: 1. Dr. Dayal R Parhi 2. Mrs. Shubhasri Kundu (filed)

Appendix-A

- Measurement of distance within underwater environment has been carried out using ultrasonic sensors externally mounted on the chassis of underwater robot. The specifications for waterproof ultrasonic sensor are given below:
  - ✓ Range: 20cm to 645cm (with partial detection up to 765-cm)
  - ✓ Resolution: 1cm
  - ✓ Supply: 3.0V to 5.5V
  - ✓ Range data output format:
    - Real time analog voltage envelope
    - Analog: (Vcc/1024) / cm
    - Serial (9600 baud rate)
  - ✓ Data refresh rate: Up to 100 milliseconds (10Hz)
  - ✓ Free run mode operation: continually measure and output range information,
  - ✓ Triggered mode operation: provides the range reading as desired
  - ✓ Sensor operating frequency: 42KHz
  - ✓ Object detection as close as 3cm from the sensor.
- The specifications for Arduino MEGA 2560 Microcontroller which receives the output from ultrasonic sensors and incorporates them in control algorithm to generate control signals towards the thrusters are given below:
  - Microcontroller: Arduino MEGA 2560 (ATmega2560).
  - Flash Memory : 256 KB.
  - Operating Voltage: 5V.
  - SRAM: 8 KB.
  - Input Voltage: 7–12V (Recommended).
  - Input Voltage (Limits): 6–20V.
  - Digital Input/ Output Pins: 54 (of Which 15 Can be Used as PWM Outputs).
  - Analog Input Pins: 16.

- The image sensor is integrated for mapping of the underwater scenario and detecting the target. The specification of image sensor are mentioned as follows:
    - ✓ Video Camera Parts: 1/3-1/4 inch.
    - ✓ CMOS Image Sensors System: PAL/CCIR NTSC/EIA
    - ✓ Scanning Frequency: PAL/CCIR: 50Hz, NTSC/EIA: 60Hz
    - ✓ Effective Pixel: PAL: 628 x 582, NTSC: 510 x 492
    - ✓ Image Area: PAL: 5.78 x 4.19mm, NTSC: 4.69 x 3.45mm
    - ✓ Minimum Illumination: 3 LUX
    - ✓ Horizontal Definition: 380 TV Lines
    - ✓ Transmission Signal: Audio, Video
    - ✓ Output Power : 50mW – 200mW
    - ✓ Power Supply: DC+ 9V – 12V, Current: 500mA
    - ✓ Output Frequency: CH1=2414; CH2=2432; CH3=2450; CH3=2468 (MHz)
- Measurement of the depth of underwater robot for its current location by integrating of on-board sonar depth sensor. The specifications for sonar depth sensor has been listed below:
    - Power Input: DC 10-18V
    - Sonar Coverage: 90
    - Detecting Range: 0.6-183 Meter
    - Sonar Frequency: wired 125KHz
    - Power Source: DC 10V-18V
    - Sonar Operating Frequency: Transducer: 200kHz / Wireless : 125kHz
    - Sonar Coverage: Transducer: 45 degree / Wireless : 90 degree
    - Depth Capability: Transducer: 328Feet (100M) / Wireless : 2-131ft (0.6-40M)
    - Depth unit: 100 M
    - Operational Wireless Frequency: 433.92MHz
    - Operational Range: 590ft or 180Meters
    - Power Input: 3.7V rechargeable lithium battery

❖ Detail description about small size underwater robot:

Compared to other available AUVs, small size and low cost underwater vehicle "GNOM Baby" has been considered here to perform real time underwater environment. The specifications of "GNOM Baby" are given in Table A. Three sets of water proof electric motor attached with propeller are attached with the hull as propulsive devices to manoeuver in surge, heave, and yaw directions only. Two of them are laterally mounted at the rear side of vehicle to control surge and yaw motions and another one is fitted vertically on the centre line of vehicle to provide heave motion along with depth control for the vehicle. On-board depth sensor of underwater robot can provide information about depth at which robot is located and five ultrasonic sensors are also embedded at surface of vehicle to measure distances between robot and obstacles or robot and target.

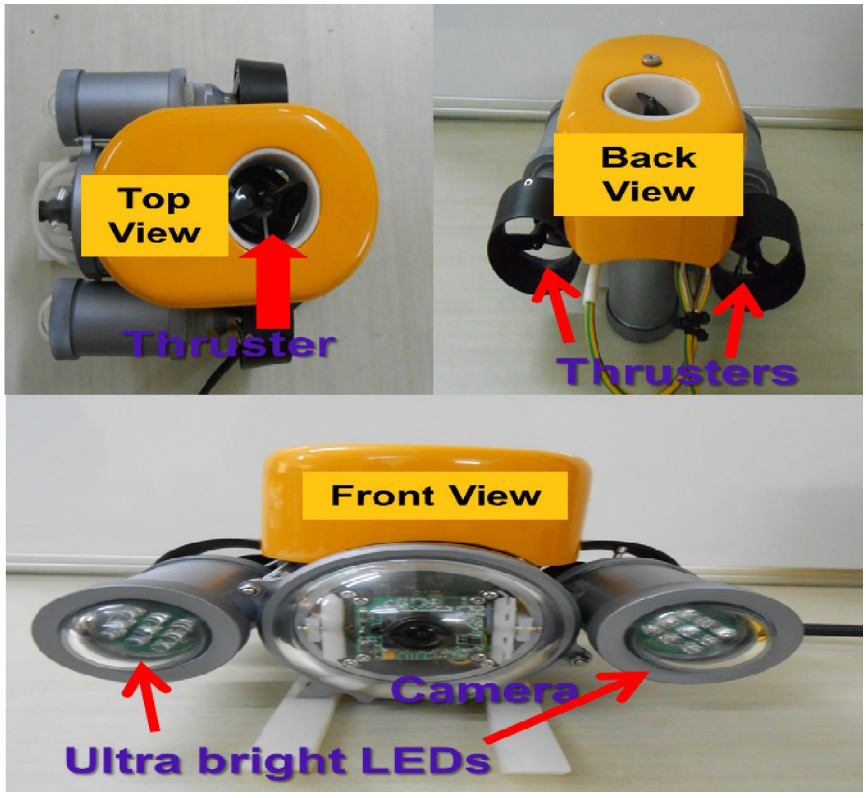


Figure A: Different Views of Underwater Robot

Online measurements by on-board sensors will be inputs for Atmega-32 microcontroller which is embedded with Arduino. The MATLAB code of navigational algorithm used in simulation mode has been interfaced with Arduino microcontroller through USB device to control the direction of motion for vehicle based on received online sensory data. Videos from camera can also be monitored at workstation in an online manner.

Table A: Specification of GNOM Baby underwater robot

No.	Items	Features
1	Thrusters (3 in number):	Horizontal (two) – speed up to 1 m/sec, Vertical (one) – speed up to 3 m/sec
2	Color camera:	PAL CCD 1/3", 450 TV Lines, 1 lux
3	Camera tilt:	Servo ±50
4	Dimensions:	210mm x185mm x150 mm
5	Ultra-bright LEDs:	Two clusters with 9 LEDs
6	Sensors:	Compass and depth sensor (on-screen overlay, auto guide mode, auto depth mode)
7	External Ultrasonic Sensors:	200Khz
8	Controller:	ATmega2560 (Arduino Mega 2560, Arduino UNO)
9	Power supply:	12-24VDC or 230VAC
10	Weight:	1.7 kg (approximately)
11	Operational depth:	10 m, (maximum depth –50m)
12	Cable:	Kevlar threads with polyethylene shielded

Appendix-B

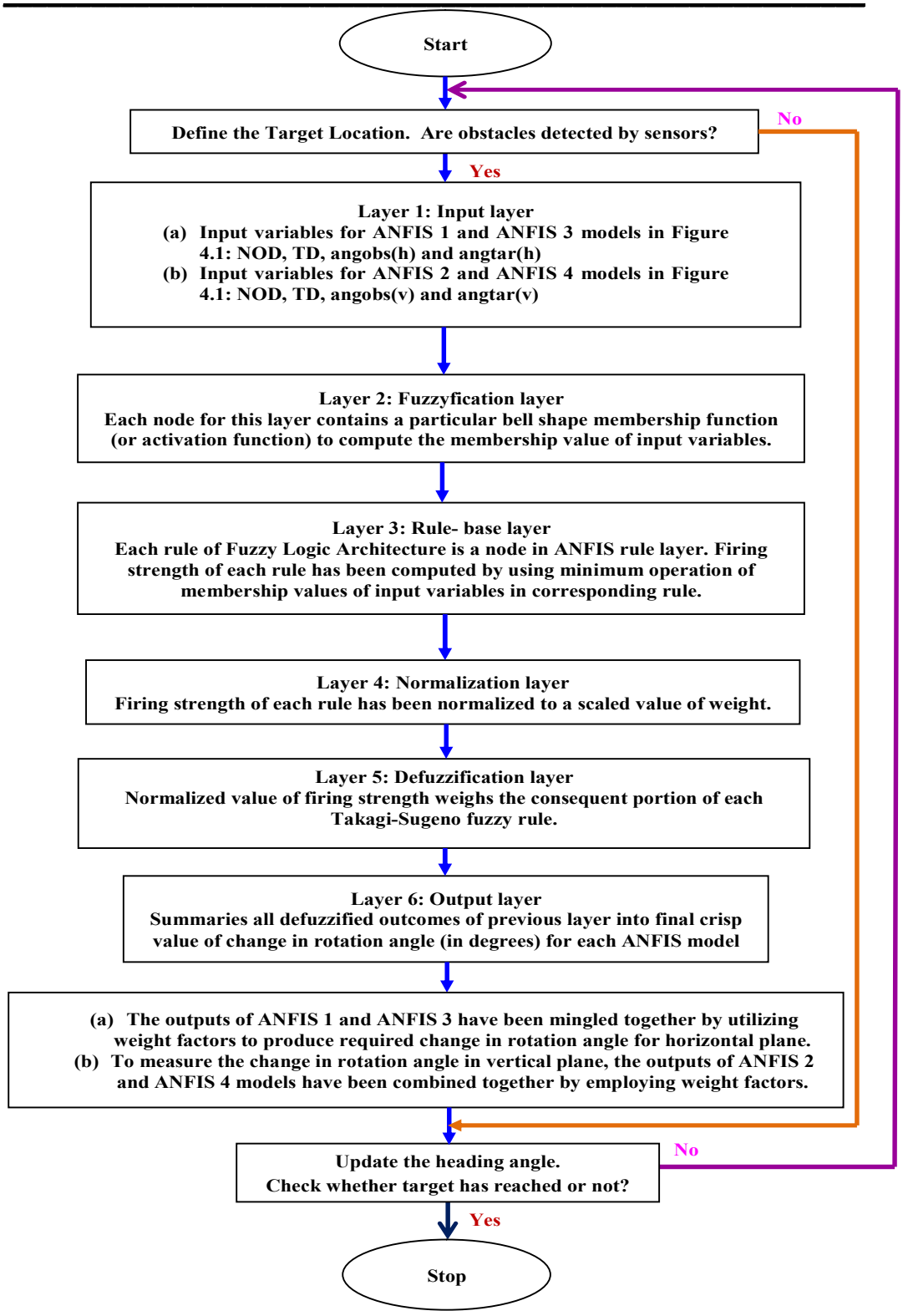


Figure B: Flow chart for ANFIS based Navigation of Underwater Robot